ORIGINAL

MICHAEL M. CARLSON (Bar No. 88048)
BRYAN J. WILSON (Bar No. 138842)
JANA G. GOLD (Bar No. 154246)
Morrison & Foerster
755 Page Mill Road
Palo Alto, California 94304-1018
Telephone: (415) 813-5600
Facsimile: (415) 494-0792

PATRICK J. FLINN (Bar No. 104423)
ALSTON & BIRD
One Atlantic Center
1201 West Peachtree Street
Atlanta, Georgia 30309
Telephone: (404) 881-7000
Facsimile: (404) 881-8777

Attorneys for Proposed Intervenor
CARO-KANN CORPORATION

120

FILED
NOV 15 3 34 PM '95
RICHARD W. WIEKING
CLERK
U.S. DISTRICT COURT
NO. DIST. OF CA. S.J.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

ROGER SCHLAFLY, Plaintiff, v. PUBLIC KEY PARTNERS and RSA DATA SECURITY, INC., Defendants.	No. CV 94 20512 SW (PVT) DECLARATION OF DR. JIMMY OMURA IN OPPOSITION TO SCHLAFLY'S MOTION AND IN SUPPORT OF CKC'S CROSS-MOTION FOR SUMMARY JUDGMENT ON THE VALIDITY OF THE STANFORD PATENTS Date: December 6, 1995 Time: 10:00 a.m. Hon. Spencer Williams

I, Jimmy K. Omura, declare:

1. I am a founder of Cylink Corporation, and I currently serve as its Chairman and as a member of its Board of Directors. I hold a Ph.D. degree from Stanford University and from 1969 to 1984 I was a professor of Electrical Engineering at the University of California in Los Angeles. I have studied the field of cryptography for 15 years. I have been awarded 12 patents from the United Stated Patent and Trademark Office for my inventions in various communications technologies. The matters stated in this declaration are of my own personal knowledge, or they represent my professional opinion and judgment, as set forth below.

2. I am familiar with the patents at issue in this suit. They represent the most fundamental advance in the field of cryptography since the invention of the alphabet substitution cipher. Before 1976, all cryptographic code schemes used a single "key" both to encode and decode a message. Two parties who wished to communicate over an open, or insecure, channel needed to exchange the single key over a different, secure channel (such as a courier) before a coded message could be sent. In 1976, Stanford University Professor Martin Hellman, with the assistance of his graduate student Whitfield Diffie, devised systems by which two parties, who could only communicate over an insecure channel, could nonetheless compute a shared secret number without the need to have a secret key delivered to both ends. Later, with the additional help of then-student Ralph Merkle, Professor Hellman developed a method of encryption in which (1) messages could be encoded with one key, and de-coded with a second key, and (2) knowledge of one key would not allow a third party to learn or obtain the other key. With the

advent of commercial computer networks this is the enabling technology for electronic commerce.

3. Under the system Hellman envisioned, each user would have two keys: a "public" key associated with the individual and known to all, and a "secret" or "private" key known only to the individual. The public and private key would be related in such a way that it would be easy to generate a public key from the private key, but "computationally infeasible" to derive the private key from the public key. Thus, a sender could encrypt a message using the recipient's public key; once the message was encrypted, however, the only way to decrypt the message would be to use the recipient's private key.

4. The application of the inventions made by Professor Hellman at Stanford University goes far beyond their use in encrypting messages. For example, the techniques associated with the Diffie-Hellman patent (discussed below) make it possible to manage keys for users on an encrypted network without requiring delivery of these secret keys. More importantly, the techniques associated with the Hellman-Merkle patent (also discussed below) make it possible to verify whether a particular message was actually sent by a particular party. If the party sending the message "signs" it by encrypting the message using her private key, then that person's public key will decrypt it. Put another way, if the sender's public key decrypts the message, then the recipient can be certain that the message was encrypted by that sender's private key. Attached to this declaration as Exhibit E is a true and correct copy of a diagram I have created showing how digital signatures work using Public Key techniques.

5. Public Key systems -- and particularly, digital signatures -- have come to have important applications with the advent of electronic commerce. Commercial transactions over electronic networks can be signed and verified, obviating the need for paper verification.

6. Attached to this Declaration as Exhibit A is a true and correct copy of United States Patent No. 4,200,770 ("Diffie-Hellman"). The patent covers what has been called "Diffie-Hellman" key exchange. The Diffie-Hellman key exchange method was conceived before any complete embodiment of Public Key, and is sometimes called a precursor to Public Key cryptography. With this invention, parties can exchange two different public numbers over an insecure channel, and together calculate a shared, secret key (Exh. A at Column 2, lines 6-22). An eavesdropper who obtains either or both public numbers would be unable to determine the parties' shared secret key (Id., Col. 5, lines 4-4). This technology is used in computer communication networks, for example, to electronically obtain keys to secure communication of users who wish to connect to the network.

7. As set forth in the patent's Abstract, the Diffie Hellman patent describes a system where each party starts out with their own secret number. Using a one-way function (that is, a function which is easy to perform but difficult to invert), each party generates a nonsecret number from their secret number. The parties then exchange their nonsecret numbers. Once the nonsecret numbers are exchanged, each party calculates the key from their retained, secret number, and the other party's exchanged nonsecret number. As described more fully in the specification of the Diffie-Hellman

patent, the exchanged nonsecret numbers will enable the parties to calculate a common key without having to exchange a key over an insecure communications channel (Exh. A, Col. 4, line 1 - Col. 5, line 41). Attached to this Declaration as Exhibit F is a true and correct copy of a chart I have drawn diagramming Diffie-Hellman key exchange and explaining the mathematics that are used in that method.

8. Attached to this declaration as Exhibit B is a true and correct copy of United States Patent No. 4,218,582 ("Hellman-Merkle"). Hellman-Merkle is the fundamental, patent covering the practice of Public Key cryptography. Hellman-Merkle includes broad claims (claims 1-6) that cover the use of Public Key regardless of the type of encryption algorithm used to generate the public-private key pairs (Exh. B, Col. 18, line 65 - Col. 20, line 48). The patent describes the use of two different keys, one for enciphering the information, and the second for deciphering the information such that the secret deciphering key is difficult to generate from the public enciphering key. (Exh. B, Col. 2, lines 48-51). Under this system, one who wishes to send a coded message uses the individual's public key to encode the message (Col. 2, lines 52-57). The recipient, using the corresponding, but secret, "private key" decodes the message (Col. 2, lines 57-59). Because the public key only works to encode the message, and only the secret private key can decode the message, the disclosure of the public key does not affect the secrecy of the message; only the holder of the private key (the recipient) can open the message.

9. The particular implementation of Public Key described in the Hellman-Merkle patent specification uses a mathematical function

known as the "knapsack problem" (Exh. B, Col. 4, lines 48-50). A variety of the problem requiring multiplicative iterations of the knapsack problem is described as the best solution the inventors had for implementing a public key system (Id., Col. 16, line 54 - Col. 18, line 60). Claims 7-17 cover various implementations of Public Key involving various forms of the knapsack problem (Id., Col. 20, line 49 - Col. 28, line 33).

10. Professor Hellman and Whitfield Diffie published two papers generally disclosing the concept of Public Key cryptography in 1976. The first was called "Multiuser Cryptographic Techniques" and was published in June 1976 as part of the proceedings of the National Computer Conference (AFIPS Conference Proceedings, Vol. 45). A true and correct copy of the "Multiuser Cryptographic Techniques" paper is attached to this declaration as Exhibit C. The second was the paper "New Directions in Cryptography," published in the journal IEEE Transactions on Information Theory, Vol. 22, No. 6 in November. A true and correct copy of the "New Directions" paper is attached to this declaration as Exhibit D.

11. The "Multiuser Cryptographic Techniques" paper suggested the concept of Public Key cryptography by proposing that the problem of key distribution could be solved by giving each user a pair of keys, one public and one private (See Exh. C at 110). The paper did not, however, disclose any particular implementation that would enable one skilled in the art to make such a system work. Indeed, the paper noted that "[a]t present, we have neither a proof that public key systems exist, nor a demonstration system" (Id. at 111).

12. The "New Directions" paper, published by the IEEE in November 1976 went only a little bit further toward disclosing the

fundamental Public Key invention. In the "New Directions" paper, Diffie and Hellman described two possible solutions to the problem of ensuring secure communications over insecure channels (See Exh. D at 647). The first possible solution was the Public Key invention -- that is a system where each user would have a public and a private key (Id. at 648). Again, however, Diffie and Hellman admitted that their only example of a possible public key cryptosystem was "[a] suggestive, although unfortunately useless example" (Id.). The second technique suggested in the "New Directions" paper was the public key distribution system set forth in the Diffie-Hellman patent (Id. at 648-49). This part of the "New Directions" paper did include a complete description of a system that would enable one skilled in the art to put together a key distribution system like the one later claimed in the Diffie-Hellman patent (Id.).

13. None of the claims of either patent consists of a purely mathematical formula. In the Diffie-Hellman patent, the only claim that includes a mathematical formula is claim 8, which claims "an apparatus for generating a secure cipher key, comprising ..." (Exh. A at Col. 11, line 21 - Col. 12, line 36). In the Hellman-Merkle patent, similarly, the first six claims do not include any mathematical formula at all, and claims 7-17 all claim either an "apparatus" or "a method" (Exh. B at Col. 20, line 49 - Col. 28, line 33). The claims of both patents are directed to methods (and apparatus) for transforming messages from one state to another, and from one party to another, not to claims for a mathematical formula.

14. The Diffie-Hellman patent does not preempt Mr. Schlafly or anyone else from using the discrete logarithm problem incorporated in claim 8, except when they use it in "an apparatus for generating

a secure cipher key" and where the algorithm is used as a means for generating the "third" and "fourth" signals described by the patent. (Exh. A, Col. 11, lines 29-37 - Col. 12, line 9) Similarly, Hellman-Merkle does not preempt the use of knapsack algorithms unless it is being used to generate a public key from a private number in a system or apparatus for communicating securely over insecure channels.

I declare under penalty of perjury that the foregoing is true and correct. Executed this 15th day of November at Sunnyvale, California.

[illegible]

Dr. Jimmy K. Omura

A

United States Patent [19]

Hellman et al.

[11] 4,200,770

[45] Apr. 29, 1980

[54] CRYPTOGRAPHIC APPARATUS AND METHOD

[75] Inventors: Martin E. Hellman, Stanford; Bailey W. Diffie, Berkeley; Ralph C. Merkle, Palo Alto, all of Calif.

[73] Assignee: Stanford University, Palo Alto, Calif.

[21] Appl. No.: 830,754

[22] Filed: Sep. 6, 1977

[51] Int. Cl.² H04L 9/04

[52] U.S. Cl. .... 178/22; 340/149 R; 375/2; 455/26

[58] Field of Search .... 178/22; 340/149 R

[56] References Cited

PUBLICATIONS

"New Directions in Cryptography", Diffie et al., *IEEE Transactions on Information Theory*, vol. IT-22, No. 6, Nov. 1976.

Diffie & Hellman, Multi-User Cryptographic Techniques", *AFIPS Conference Proceedings*, vol. 45, pp. 109-112, Jun. 8, 1976.

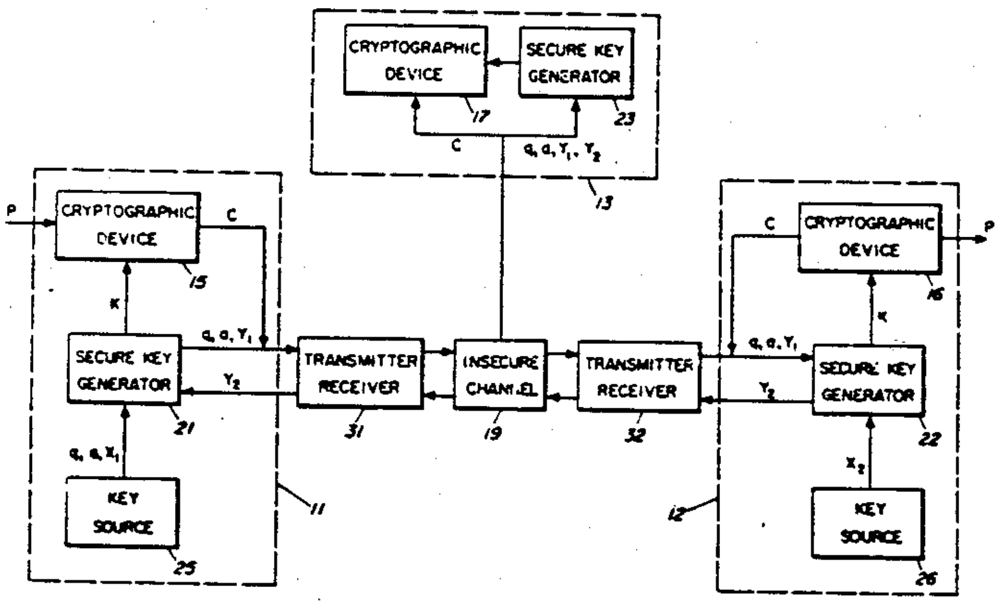
Primary Examiner—Howard A. Birmiel

Attorney, Agent, or Firm—Flehr, Hohbach, Test

[57] ABSTRACT

A cryptographic system transmits a computationally secure cryptogram over an insecure communication channel without prearrangement of a cipher key. A secure cipher key is generated by the conversers from transformations of exchanged transformed signals. The conversers each possess a secret signal and exchange an initial transformation of the secret signal with the other converser. The received transformation of the other converser's secret signal is again transformed with the receiving converser's secret signal to generate a secure cipher key. The transformations use non-secret operations that are easily performed but extremely difficult to invert. It is infeasible for an eavesdropper to invert the initial transformation to obtain either conversers' secret signal, or duplicate the latter transformation to obtain the secure cipher key.

8 Claims, 6 Drawing Figures



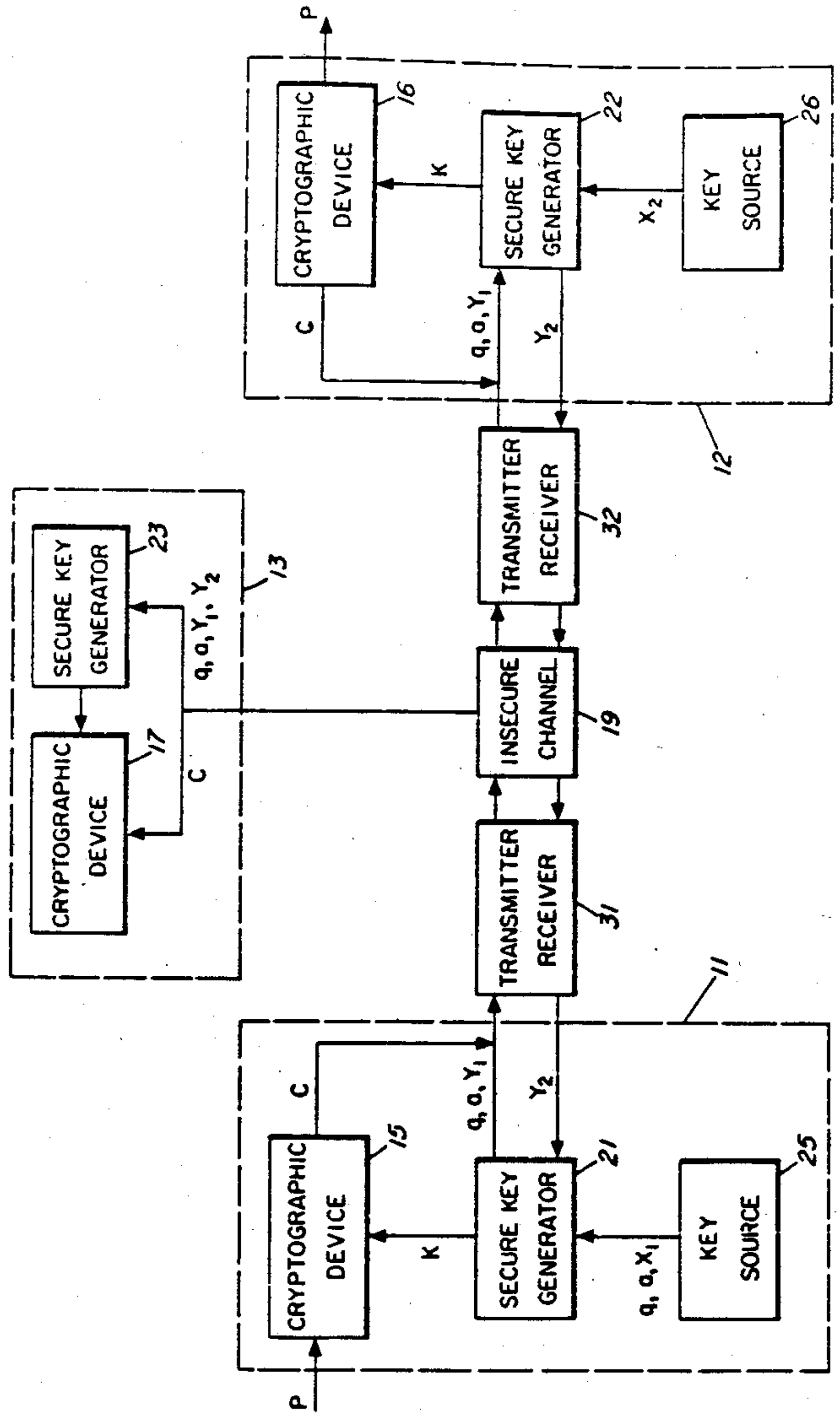

P
CRYPTOGRAPHIC DEVICE
16
K
SECURE KEY GENERATOR
22
X_2
KEY SOURCE
26
C
q, a, Y_1
Y_2
12
TRANSMITTER RECEIVER
32
INSECURE CHANNEL
19
TRANSMITTER RECEIVER
31
SECURE KEY GENERATOR
23
CRYPTOGRAPHIC DEVICE
17
q, a, Y_1, Y_2
C
13
11
C
q, a, Y_1
Y_2
CRYPTOGRAPHIC DEVICE
15
K
SECURE KEY GENERATOR
21
q, a, X_1
KEY SOURCE
25
P

FIG. 1

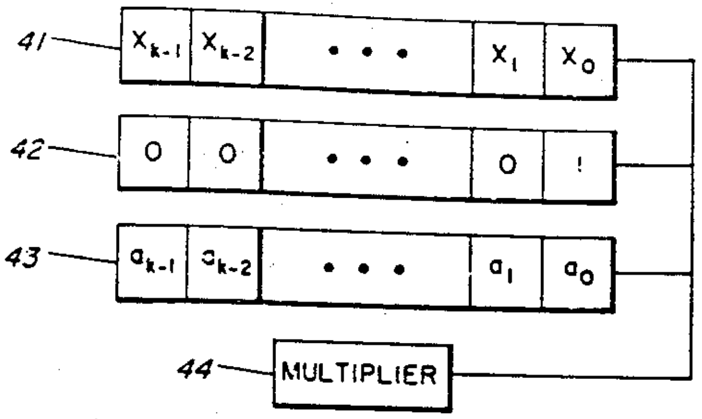

41
X_{k-1}
X_{k-2}
• • •
X_1
X_0
42
0
0
• • •
0
1
43
a_{k-1}
a_{k-2}
• • •
a_1
a_0
44
MULTIPLIER

FIG. 2

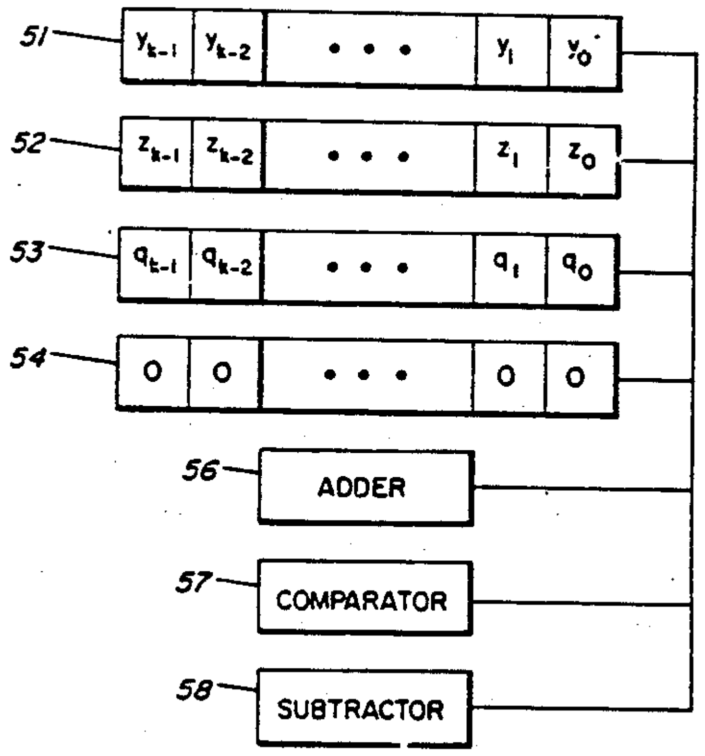

51
y_{k-1}
y_{k-2}
• • •
y_1
y_0
52
z_{k-1}
z_{k-2}
• • •
z_1
z_0
53
q_{k-1}
q_{k-2}
• • •
q_1
q_0
54
0
0
• • •
0
0
56
ADDER
57
COMPARATOR
58
SUBTRACTOR

FIG. 3

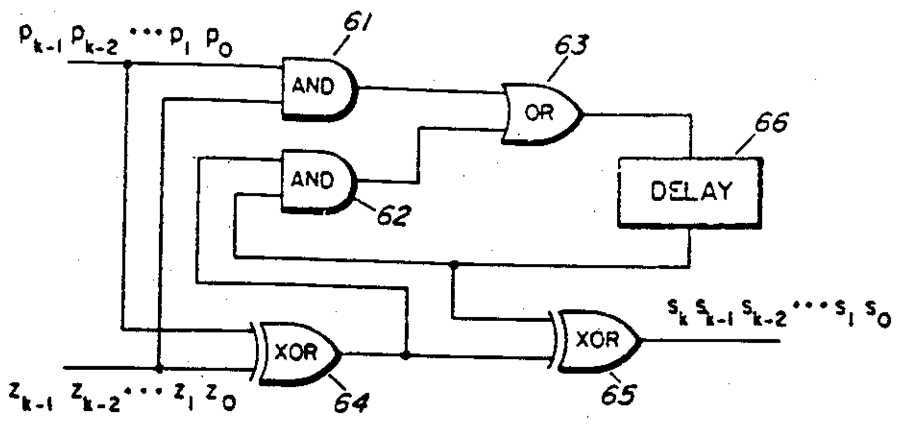

$p_{k-1}\ p_{k-2} \cdots p_1\ p_0$
61
AND
63
OR
66
DELAY
AND
62
XOR
64
XOR
65
$s_k\ s_{k-1}\ s_{k-2} \cdots s_1\ s_0$
$z_{k-1}\ z_{k-2} \cdots z_1\ z_0$

FIG. 4

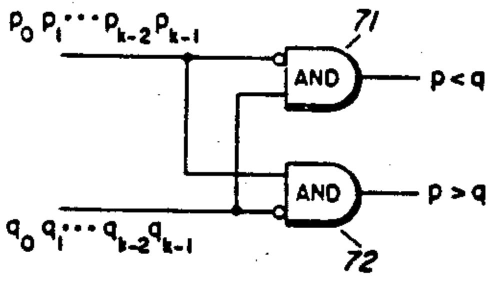

$p_0\ p_1 \cdots p_{k-2}\ p_{k-1}$
71
AND
$p<q$
AND
$p>q$
$q_0\ q_1 \cdots q_{k-2}\ q_{k-1}$
72

FIG. 5

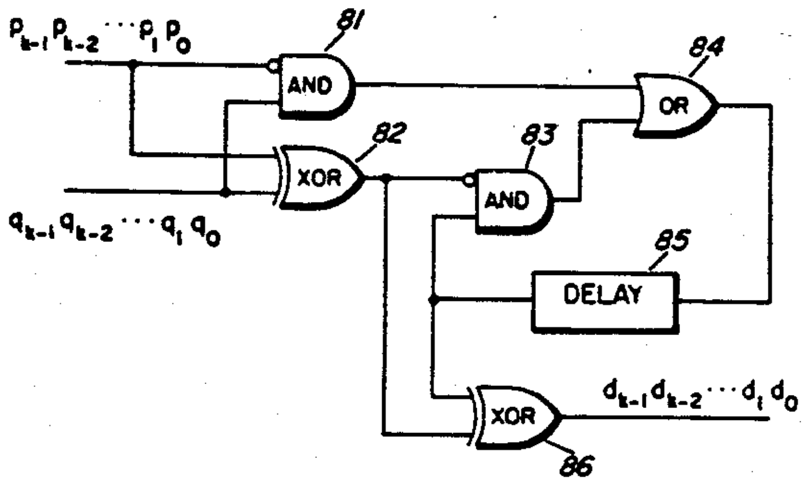

$p_{k-1}\ p_{k-2} \cdots p_1\ p_0$
81
AND
84
OR
82
XOR
83
AND
$q_{k-1}\ q_{k-2} \cdots q_1\ q_0$
85
DELAY
XOR
86
$d_{k-1}\ d_{k-2} \cdots d_1\ d_0$

FIG. 6

CRYPTOGRAPHIC APPARATUS AND METHOD

The Government has rights in this invention pursuant to Grant No. ENG-10173 of the National Science Foundation and IPA No. 0005.

BACKGROUND OF THE INVENTION

1. Field of Invention

The invention relates to cryptographic systems.

2. Description of Prior Art

Cryptographic systems are widely used to ensure the privacy and authenticity of messages communicated over insecure channels. A privacy system prevents the extraction of information by unauthorized parties from messages transmitted over an insecure channel, thus assuring the sender of a message that it is being read only by the intended receiver. An authentication system prevents the unauthorized injection of messages into an insecure channel, assuring the receiver of the message of the legitimacy of its sender.

One of the principal difficulties with existing cryptographic systems is the need for the sender and receiver to exchange a cipher key over a secure channel to which the unauthorized party does not have access. The exchange of a cipher key frequently is done by sending the key in advance over a secure channel such as private courier or registered mail; such secure channels are usually slow and expensive.

Diffie, et al, in "Multiuser Cryptographic Techniques," *AFIPS—Conference Proceedings*, Vol. 45, pp. 109-112, June 8, 1976, propose the concept of a public key cryptosystem that would eliminate the need for a secure channel by making the sender's keying information public. It is also proposed how such a public key cryptosystem could allow an authentication system which generates an unforgeable message dependent digital signature. Diffie presents the idea of using a pair of keys E and D, for enciphering and deciphering a message, such that E is public information while D is kept secret by the intended receiver. Further, although D is determined by E, it is infeasible to compute D from E. Diffie suggests the plausibility of designing such a public key cryptosystem that would allow a user to encipher a message and send it to the intended receiver, but only the intended receiver could decipher it. While suggesting the plausibility of designing such systems, Diffie presents neither proof that public key cryptosystems exist, nor a demonstration system.

Diffie suggests three plausibility arguments for the existence of a public key cryptosystem; a matrix approach, a machine language approach and a logic mapping approach. While the matrix approach can be designed with matrices that require a demonstrably infeasible cryptanalytic time (i.e., computing D from E) using known methods, the matrix approach exhibits a lack of practical utility because of the enormous dimensions of the required matrices. The machine language approach and logic mapping approach are also suggested, but there is not way shown to design them in such a manner that they would require demonstrably infeasible cryptanalytic time.

Diffie also introduces a procedure using the proposed public key cryptosystems, that could allow the receiver to easily verify the authenticity of a message, but which prevents him from generating apparently authenticated messages. Diffie describes a protocol to be followed to obtain authentication with the proposed public key

cryptosystem. However, the authentication procedure relies on the existence of a public key cryptosystem which Diffie did not provide.

SUMMARY AND OBJECTS OF THE INVENTION

Accordingly, it is an object of this invention to allow authorized parties to a conversation (conversers) to converse privately even though an unauthorized party (eavesdropper) intercepts all of their communications.

Another object of this invention is to allow conversers on an insecure channel to authenticate each other's identity.

An illustrated embodiment of the present invention describes a method for communicating securely over an insecure channel without prearrangement of a cipher key. A secure cipher key is generated from transformations of exchanged transformed signals, which exchanged transformed signals are easy to effect but difficult to invert. The generated secure cipher key is used to encipher and decipher messages transmitted over the insecure communication channel.

This illustrated embodiment of the present invention describes a method and apparatus for generating a secure cipher key for use with conventional cryptographic communication between conversers over an insecure channel. The illustrated embodiment differs from a public key cryptosystem in that it provides a secure cipher key that is used with a conventional cryptographic system; a public key cryptosystem does not require a conventional cryptographic system. Further, the illustrated embodiment provides a means of transforming a signal that is practical to implement and is demonstrably infeasible to invert using known methods.

In the present invention a first converser transforms, in a manner infeasible to invert, a first signal while a second converser transforms, also in a manner infeasible to invert, a second signal. The first converser transmits the transformed first signal to the second converser, keeping the first signal secret, and the second converser transmits the transformed second signal to the first converser, keeping the second signal secret. The first converser then transforms the first signal with the transformed second signal to generate a third signal, representing a secure cipher key, that is infeasible to generate solely by transforming the transformed first signal and the transformed second signal. And, the second converser transforms the second signal with the transformed first signal to generate a fourth signal, also representing the secure cipher key, that is infeasible to generate solely by transforming the transformed first signal and the transformed second signal.

Another illustrated embodiment of the present invention describes a method for allowing a converser to authenticate another converser's identity. A first converser transforms, in a manner infeasible to invert, a first signal while a second converser transforms, also in a manner infeasible to invert, a second signal. The second converser places the transformed second signal in a public directory as the second converser's means of identification, keeping the second signal secret. The first converser transmits the transformed first signal to the second converser, with whom the first converser desires to communicate, keeping the first signal secret. The first converser then transforms the first signal with the second converser's transformed second signal, obtained from the public directory, to generate a third signal. The third signal represents a secure cipher key to be used for conventional cryptographic communication with the second converser, that is infeasible to generate solely by transforming the transformed first signal and the transformed second signal. The second converser transforms the second signal with the transformed first signal to generate a fourth signal, also representing the secure cipher key, that is infeasible to generate solely by transforming the transformed first signal and the transformed second signal. The second converser's identity is authenticated by the first converser by the second converser's ability to generate the fourth signal, representing the secure cipher key, and to use the secure cipher key in communicating over a conventional cryptographic system.

Additional objects and features of the present invention will appear from the description that follows wherein the preferred embodiments have been set forth in detail in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of a cryptographic system that transmits a computationally secure cryptogram over an insecure communication channel.

FIG. 2 is a block diagram of a secure key generator for raising various numbers to various powers in modulo arithmetic.

FIG. 3 is a block diagram of a multiplier for performing multiplications in the secure key generator of FIG. 2.

FIG. 4 is a detailed schematic diagram of an adder for performing additions in the multiplier of FIG. 3.

FIG. 5 is a detailed schematic diagram of a comparator for performing magnitude comparisons in the multiplier of FIG. 3.

FIG. 6 is a detailed schematic diagram of a subtractor for performing subtractions in the multiplier of FIG. 3.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Referring to FIG. 1, a cryptographic system is shown in which all communications take place over an insecure communication channel 19, for example a telephone line. Two-way communication is exchanged on the insecure channel 19 between converser 11 and converser 12 using transmitter/receivers 31 and 32, for example modems such as Bell 210 modems. Converser 11 possesses an unenciphered or plaintext message P to be communicated to converser 12. Converser 11 and converser 12 include cryptographic devices 15 and 16 respectively, for enciphering and deciphering information under the action of a cipher key K on line K. For example, the cryptographic devices 15 and 16 may include the recently adopted National Data Encryption Standard. The cryptographic devices 15 and 16 implement transformations S_K and S_K^{-1} (the transformation which is the inverse of S_K) when loaded with key K. For example, key K may be a sequence of random letters or digits. The cryptographic device 15 enciphers the plaintext message P into an enciphered message or ciphertext C on line C that is transmitted by converser 11 through the insecure channel 19; the ciphertext C is received by converser 12 and deciphered by cryptographic device 16 to obtain the plaintext message P. An unauthorized party or eavesdropper 13 is assumed to have a cryptographic device 17 and to have access to the insecure channel 19, so if he knew the key K he could decipher the ciphertext C to obtain the plaintext message P.

Converser 11 and converser 12 include independent key sources 25 and 26 respectively, which generate numbers or signals that represent numbers. For example, the key sources may be random number generators that are implemented from noisy amplifiers (e.g., Fairchild μ709 operational amplifiers) with a polarity detector. Key source 25 generates three signals, q, a, and X_1, and key source 26 generates X_2; a, X_1 and X_2 may be signals that represent independent random numbers chosen uniformly from the set of integers (1, 2, . . . $q-1$). Signals q and a are transmitted to the secure key generator 21 and are transmitted through the insecure channel 19 to secure key generator 22. Signals X_1 and X_2 are kept secret by converser 11 and converser 12 respectively, are given to the secure key generators 21 and 22 respectively, but are not transmitted through the insecure channel 19.

Converser 11 and converser 12 also include secure key generators 21 and 22 respectively, which accept the signals generated by the respective key sources 25 and 26. Secure key generator 22 also receives the signals q and a which are transmitted through the insecure channel 19. The secure key generators 21 and 22 generate signals Y_1 and Y_2 respectively by transforming X_1 and X_2 respectively with signals q and a in a manner that is easily performed but extremely difficult or infeasible to invert. A task is considered infeasible if its cost as measured by either the amount of memory used or the computing time is finite but impossibly large, for example, on the order of approximately 10^{30} operations with existing computational methods and equipment such transformations are characterized by the class of mathematical functions known as one-way cipher functions.

Signal Y_1 may be generated to represent the number obtained by raising the number represented by signal a to the power represented by signal X_1, modulo the number represented by signal q; this transformation may be represented symbolically as $Y_1 = a^{X_1} \bmod q$. Signal Y_2 may be generated to represent the number obtained by raising the number represented by signal a to the power represented by signal X_2, modulo the number represented by signal q; this transformation may be represented symbolically as $Y_2 = a^{X_2} \bmod q$.

Signals Y_1 and Y_2 are exchanged by transmitting Y_1 and Y_2 through the insecure channel 19 to secure key generators 22 and 21 respectively. Secure key generator 21 then generates a secure key K by transforming signal Y_2 with signals q, a and X_1, and secure key generator 22 generates the same secure key K by transforming Y_1 with signals q, a and X_2.

Secure key generator 21 may generate a secure key K represented by the number obtained by raising the number represented by signal Y_2 to the power represented by signal X_1, modulo the number represented by signal q; this transformation may be represented symbolically as

$$K = Y_2^{X_1} \bmod q = (a^{X_2})^{X_1} \bmod q = a^{X_1 X_2} \bmod q.$$

Secure key generator 22 may also generate the same secure key K represented by the number obtained by raising the number represented by signal Y_1 to the power represented by signal X_2, modulo the number represented by signal q; this transformation may be represented symbolically as

$$K = Y_1^{X_2} \bmod q = (a^{X_1})^{X_2} \bmod q = a^{X_1 X_2} \bmod q.$$

Conversers 11 and 12 then have the same secure key K which may be used with cryptographic devices 15 and 16.

The eavesdropper 13 is assumed to have a secure key generator 23 and to have access to all signals transmitted through the insecure channel 19, including signals q, a, Y_1, and Y_2. The difficulty of inverting the transformations which generated signals Y_1 and Y_2 make it infeasible for the eavesdropper 13 to generate signals X_1 or X_2. Further, the secure key K is infeasible to generate solely with signals q, a, Y_1 and Y_2.

The eavesdropper 13 is unable to compute the secure key K by multiplication or exponentiation; multiplication yields

$$Y_1Y_2=a^{X_1+X_2} \bmod q \neq K$$

and exponentiation yields either

$$Y_1^{Y_2}=a^{(X_1a^{X_2})} \neq K$$

or

$$Y_2^{Y_1}=a^{(X_2a^{X_1})} \neq K.$$

The eavesdropper in theory could obtain X_1 or X_2 from q, a and Y_1 and Y_2 by raising a to the first, second, third, etc., powers until Y_1 or Y_2 was obtained. This is prevented by choosing q to be a large number; if q is a 200 bit quantity, the average number of trials before success is on the order of $2^{198}=4\times10^{59}$ and is physically infeasible. Improved algorithms for computing logarithms modulo q (if $Y=a^X$ mod q, X is the logarithm of Y to the base a modulo q) are known but, if $q=2r+1$ with q and r being prime, then the most efficient known algorithm requires approximately $q^{\frac{1}{2}}$ operations. Again, taking $q=2^{200}$, about $2^{100}=10^{30}$ operations are required, still physically infeasible. An example of such a paid is $r=(2^{121}\cdot5^2\cdot7^2\cdot11^2\cdot13\cdot17\cdot19\cdot23\cdot29\cdot31\cdot37\cdot41\cdot43\cdot47\cdot53\cdot59)+1$ and $q=2r+1$. Other restrictions on q, a, X_1 and X_2 may also be imposed.

The secure key generators 21 and 22, for raising various numbers to various powers modulo q, can be implemented in electronic circuitry as shown in FIG. 2. For ease of illustration, FIG. 2 depicts raising a to the X power modulo q; raising Y to the X power modulo q is obtained by initially loading Y, instead of a, into the A register 43.

FIG. 2 shows the initial contents of three registers 41, 42 and 43. The binary representation of X ($x_{k-1}\,x_{k-2}\ldots x_1x_0$) is loaded into the X register 41; 1 is loaded into the R register 42; and, the binary representation of a is loaded into the A register 43, corresponding to i=0. The number of bits k in each register is the least integer such that $2^k \geqq q$. If k=200, then all three registers can be obtained from a single 1024 bit random access memory (RAM) such as the Intel 2102. The implementation of multiplier 44, for multiplying two numbers modulo q, will be described in more detail later.

Referring to FIG. 2, if the low order bit, containing x_0, of the X register 41 equals 1 then the R register 42 and the A register 43 contents are multiplied modulo q and their product, also a k bit quantity, replaces the contents of the R register 42. If $x_0=0$, the R register 42 contents are left unchanged. In either case, the A register 43 is then loaded twice into the multiplier 44 so that the square, modulo q, of the A register 43 contents is computed. This value, $a^{(2^{i+1})}$, replaces the contents of the A register 43. The X register 41 contents are shifted one bit to the right and a 0 is shifted in at the left so its contents are now $0x_{k-1}x_{k-2}\ldots x_2x_1$.

The low order bit, containing x_1, of the X register 41 is examined. If it equals 1 then, as before, the R register 42 and A register 43 contents are multiplied modulo q and their product replaces the contents of the R register 42. If the low order bit equals 0 then the R register 42 contents are left unchanged. In either case, the contents of the A register 43 are replaced by the square, modulo q, of the previous contents. The X register 41 contents are shifted one bit to the right and a 0 is shifted in at the left so its contents are now $00x_{k-1}\,x_{k-2}\ldots x_3x_2$.

This process continues until the X register 41 contains all 0's, at which point the value of a^X modulo q is stored in the R register 42.

An example is helpful in following this process. Taking q=23, we find k=5 from $2^k \geqq q$. If a=7 and X=18 then $a^X=7^{18}=1{,}628{,}413{,}597{,}910{,}449=23(70{,}800{,}591{,}213{,}497)+18$ so a^X modulo q equals 18. This straightforward but laborious method of computing a^X modulo q is used as a check to show that the method of FIG. 2, shown below, yields the correct result. The R register 42 and A register 43 contents are shown in decimal form to facilitate understanding.

i	X (in binary)	R	A
0	10010	1	7
1	01001	1	3
2	00100	3	9
3	00010	3	12
4	00001	3	6
5	00000	18	13

The row marked i=0 corresponds to the initial contents of each register, X=18, R=1 and A=a=7. Then, as described above, because the right most bit of X register 41 is 0, the R register 42 contents are left unchanged, the contents of the A register 43 are replaced by the square, modulo 23, of its previous contents ($7^2=49=2\times23+3=3$ modulo 23), the contents of the X register 41 are shifted one bit to the right, and the process continues. Only when i=1 and 4 do the rightmost bit of the X register 41 contents equal 1, so only going from i=1 to 2 and from i=4 to 5 is the R register 42 replaced by RA modulo q. When i=5, X=0 so the process is complete and the result, 18, is in the R register 42.

Note that the same result, 18, is obtained here as in the straightforward calculation of 7^{18} modulo 23, but that here large numbers never resulted.

Another way to understand the process is to note that the A register contains a, a^2, a^4, a^8 and a^{16} when i=0, 1, 2, 3 and 4 respectively, and that $a^{18}=a^{16}\,a^2$, so only these two values need to be multiplied.

FIG. 3 continues the description of this illustrative implementation by depicting an implementation of the modulo q multiplier 44 in FIG. 2. The two numbers, y and z, to be multiplied are loaded into the Y and Z registers 51 and 52 respectively, and q is loaded in the Q register 53. The product yz modulo q will be produced in the P register 54 which is initially set to 0. If k=200, then all four registers can be obtained from a single 1024 bit RAM such as the Intel 2102. The implementation of FIG. 3 is based on the fact that y z mod q=y_0z mod q+$2y_1z$ mod q+$4y_2z$ mod q+ . . . +$2^{k-1}y_{k-1}z$ mod q, where $y_{k-1}y_{k-2} \ldots y_1y_0$ is the binary representation of Y.

To multiply y times z, if the rightmost bit, containing y_0, of the Y register 51 is 1 then the contents of the Z register 53 are added to the P register 54 by adder 55. If $y_0=0$, then the P register 54 is unchanged. Then the Q and P register contents are compared by comparator 56 to determine if the contents of the P register 54 are greater than or equal to q, the contents of the Q register 53. If the contents of the P register 54 are greater than or equal to q then subtractor 57 substracts q from the contents of the P register 54 and places the difference in the P register 54, if less than q the P register 54 is unchanged.

Next, the contents of Y register 51 are shifted one bit to the right and a 0 is shifted in at the left so its contents become $0y_{k-1}\ y_{k-2} \ldots y_2y_1$, so that y_1 is ready for computing $2y_1z$ mod q. The quantity 2z mod q is computed for this purpose by using adder 55 to add z to itself, using comparator 56 to determine if the result, 2z, is less than q, and using subtractor 57 for subtracting q from 2z if the result is not less than q. The result, 2z mod q is then stored in the Z register 52. The rightmost bit, containing y_1, of the Y register 51 is then examined, as before, and the process repeats.

This process is repeated a maximum of k times or until the Y register 51 contains all 0's, at which point xy modulo q is stored in the P register 54.

As an example of these operations, consider the problem of computing 7×7 modulo 23 needed to produce the second state of the A register when 7^{13} mod 23 was computed. The following steps show the successive contents of the Y, Z and P registers which result in the answer 7×7=3 modulo 23.

i	Y (in binary)	Z	P
0	00111	7	0
1	00011	14	0 + 7 = 7
2	00001	5	7 + 14 = 21
3	00000	10	21 + 5 = 3 mod 23

FIG. 4 depicts an implementation of an adder 55 for adding two k bit numbers p and z. The numbers are presented one bit at a time to the device, low order bit first, and the delay element is initially set to 0. (The delay represents the binary carry bit.) The AND gate 61 determines if the carry bit should be a one based on p_i and z_i both being 1 and the AND gate 62 determines if the carry should be a 1 based on the previous carry being a 1 and one of p_i or z_i being 1. If either of these two conditions is met, the OR gate 63 has an output of 1 indicating a carry to the next stage. The two exclusive-or (XOR) gates 64 and 65 determine the i^{th} bit of the sum, s_i, as the modulo-2 sum of p_i, z_i and the carry bit from the previous stage. The delay 66 stores the previous carry bit. Typical parts for implementing these gates and the delay are SN7400, SN7404, and SN7474.

FIG. 5 depicts an implementation of a comparator 56 for comparing two numbers p and q. The two numbers are presented one bit at a time, high order bit first. If neither the p<q nor the p>q outputs have been triggered after the last bits p_0 and q_0 have been presented, then p=q. The first triggering of either the p<q or the p>q output causes the comparison operation to cease. The two AND gates 71 and 72 each have one input inverted (denoted by a circle at the input). An SN7400 and SN7404 provide all of the needed logic circuits.

FIG. 6 depicts an implementation of a subtractor 57 for subtracting two numbers. Because the numbers subtracted in FIG. 3 always produce a non-negative difference, there is no need to worry about negative differences. The larger number, the minuend, is labelled p and the smaller number, the substrahend, is labelled q. Both p and q are presented serially to the subtractor 57, low order bit first. AND gates 81 and 83, OR gate 84 and XOR gate 82 determine if borrowing (negative carrying) is in effect. A borrow occurs if either $p_i=0$ and $q_i=1$, or $p_i=q_i$ and borrowing occurred in the previous stage. The delay 85 stores the previous borrow state. The i^{th} bit of the difference, d_i, is computed as the XOR, or modulo-2 difference, of p_i, q_i and the borrow bit. The output of XOR gate 82 gives the modulo-2 difference between p_i and q_i, and XOR gate 86 takes the modulo-2 difference of this with the previous borrow bit. Typical parts for implementing these gates and the delay are SN7400, SN7404 and SN7474.

There are many methods for implementing this form of the invention. The signals q and a may be public knowledge rather than generated by the key source 25. Further, it should be appreciated that the present invention has the capability of being modified by the use of additional transformations or exchanges of signals.

In some applications, it will prove valuable to have the i^{th} converser on the system generate Y_i as above and place it in a public file or directory rather than transmitting it to another converser with whom he wishes to communicate. Then two conversers i and j who wish to establish a secure channel will use $K_{ij}=Y_i^{X_j}$ mod $q=Y_j^{X_i}$ mod q as their key. The advantage is that converser i, having once proved his identity to the system through the use of his driver's license, fingerprint, etc., can prove his identity to converser j by his ability to compute K_{ij} and encrypt data with it.

Variations on the above described embodiment are possible. For example, in the above method based on logarithms modulo q, m-dimensional vectors, each of whose components are between 0 and q−1 could also be used. Then all operations are performed in the finite field with q^m elements, which operations are well described in the literature. Thus, although the best mode contemplated for carrying out the present invention has been herein shown and described, it will be apparent that modification and variation may be made without departing from what is regarded to be the subject matter of this invention.

What is claimed is:

1. A secure key generator comprising:

a first input connected to receive an applied first signal;

a second input connected to receive an applied second signal;

a first output;

a second output; and

means for generating at the first output a third signal, that is a transformation of said first signal and which transformation is infeasible to invert, and for generating at the second output a fourth signal, that is a transformation of said second signal with said first signal, which represents a secure key and is infeasible to generate solely with said second signal and said third signal.

2. In a method of communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

generating and transforming, in a manner infeasible to invert, a first signal at the transmitter to generate a transformed first signal;

generating and transforming, in a manner infeasible to invert, a second signal at the receiver to generate a transformed second signal;

transmitting said transformed first signal from the transmitter to the receiver;

transmitting said transformed second signal from the receiver to the transmitter;

transforming said transformed second signal with said first signal at the transmitter to generate a third signal, representing a secure cipher key, that is infeasible to generate solely with said transformed first signal and said transformed second signal;

transforming said transformed first signal with said second signal at the receiver to generate a fourth signal that is identical to the third signal and represents said secure cipher key;

enciphering the message with said secure cipher key at the transmitter;

transmitting the enciphered message from the transmitter to the receiver; and

deciphering the enciphered message with said secure cipher key at the receiver.

3. In a method of communicating securely over an insecure communication channel as in claim 2, further comprising:

authenticating the receiver's identity at the transmitter from the receiver's ability to generate the fourth signal, representing the secure cipher key.

4. A method of generating a secure cipher key between a transmitter and receiver comprising the steps of:

generating and transforming, in a manner infeasible to invert, a first signal at the transmitter to generate a transformed first signal;

generating and transforming, in a manner infeasible to invert, a second signal at the receiver to generate a transformed second signal;

transmitting said transformed first signal from the transmitter to the receiver

transmitting said transformed second signal from the receiver to the transmitter;

transforming said transformed second signal with said first signal at the transmitter to generate a third signal, representing a secure cipher key, that is infeasible to generate solely with said transformed first signal and said transformed second signal; and

transforming said transformed first signal with said second signal at the receiver to generate a fourth signal that is identical to the third signal and represents said secure cipher key.

5. An apparatus for generating a secure cipher key comprising:

a first secure key generator having a first input connected to receive an applied first signal, having a second input connected to receive a second signal, having a first and second outputs, and having a means for generating at the first output a third signal, that is a transformation of said first signal and which transformation is infeasible to invert, and for generating at the second output a fourth signal, that is a transformation of said second signal with said first signal, which represents a secure key and is infeasible to generate solely with said second signal and said third signal; and

a second secure key generator having a first input connected to receive an applied fifth signal, having a second input connected to receive said third signal, having a first and second outputs, and having a means for generating at the first output said second signal, that is a transformation of said fifth signal and which transformation is infeasible to invert, and for generating at the second output a sixth signal, that is a transformation of said third signal with said fifth signal, which represents the secure key and is infeasible to generate solely with said second signal and said third signal.

6. A method of generating a secure cipher key between a transmitter and receiver comprising the steps of:

transforming, in a manner infeasible to invert, a first signal at the transmitter to generate a transformed first signal wherein transforming said first signal is performed by raising a first number to a power represented by said first signal, modulo a second number;

transforming, in a manner infeasible to invert, a second signal at the receiver to generate a transformed second signal, wherein transforming said second signal is performed by raising the first number to a power represented by said second signal, modulo the second number;

transmitting said transformed first signal from the transmitter to the receiver;

transmitting said transformed second signal from the receiver to the transmitter;

transforming said transformed second signal with said first signal at the transmitter to generate a third signal, representing a secure cipher key, that is infeasible to generate solely with said transformed first signal and said transformed second signal, wherein transforming said transformed second signal with said first signal is performed by raising a number represented by said transformed second signal to a power represented by said first signal, modulo the second number; and

transforming said transformed first signal with said second signal at the receiver to generate a fourth signal, representing said secure cipher key, that is infeasible to generate solely with said transformed first signal and said transformed second signal, wherein transforming said transformed first signal with said second signal is performed by raising a number represented by said transformed first signal to a power represented by said second signal, modulo the second number.

7. An apparatus for generating a secure cipher key comprising:

a first secure key generator having a first input connected to receive an applied first signal, having a second input connected to receive a second signal, having first and second outputs, and having a means for generating at the first output a third signal, that is a transformation of said first signal in which said transformation includes raising a first number to a power represented by said first signal, modulo or second number, and for generating at the second output a fourth signal, that is a transformation of said second signal with said first signal which transformation includes raising a number represented by said second signal to a power represented by said first signal, modulo the second number, which represents a secure key and is infeasible

P018897

to generate solely with said second signal and said third signal; and

a second secure key generator having a first input connected to receive an applied fifth signal, having a second input connected to receive said third signal, having a first and second outputs, and having a means for generating at the first output said second signal, that is a transformation of said fifth signal in which said transformation includes raising a first number to a power represented by said fifth signal, modulo the second number, and for generating at the second output a sixth signal, that is a transformation of a said third signal with said fifth signal which transformation includes raising a number represented by said third signal to a power represented by said fifth signal, modulo the second number, which represents the secure key and is infeasible to generate solely with said second signal and said third signal.

8. An apparatus for generating a secure cipher key comprising:

a first secure key generator having a first input connected to receive an applied first signal, having a second input connected to receive a second signal, having a first and second outputs, and having a means for generating at the first output a third signal, said third signal Y_i being described by

$$Y_i = a^{X_i} \bmod q$$

where

q = a large prime number

a = a random number, such that $1 \leqq a \leqq q-1$

x_i = the first signal which represents a random number, such that $1 \leqq X_i \leqq q-1$

a transformation of said first signal which is infeasible to invert, and for generating at the second output a fourth signal, said fourth signal K_{ij} being described by

$$K_{ij} = Y_j^{X_i} \bmod q$$

where

Y_j = the second signal

a transformation of said second signal with said first signal, which represents said secure cipher key and is infeasible to generate solely with said second signal and said third signal; and

a second secure key generator having a first input connected to receive an applied fifth signal, having a second input connected to receive said third signal, having a first and second outputs, and having a means for generating at the first output a second signal, said second signal Y_j being described by

$$Y_j = a^{X_j} \bmod q$$

where

X_j = the fifth signal which represents a random number, such that $1 \leqq X_j \leqq q-1$

a transformation of said fifth signal which is infeasible to invert, and for generating at the second output a sixth signal, said sixth signal K_{ij} being described by

$$K_{ij} = Y_i^{X_j} \bmod q$$

a transformation of said third signal with said fifth signal, which represents the secure key and is infeasible to generate solely with said second signal and said third signal.

* * * * *

P018898

B

United States Patent [19]
Hellman et al.

[11] **4,218,582**
[45] **Aug. 19, 1980**

[54] PUBLIC KEY CRYPTOGRAPHIC APPARATUS AND METHOD

[75] Inventors: **Martin E. Hellman**, Stanford; **Ralph C. Merkle**, Palo Alto, both of Calif.

[73] Assignee: **The Board of Trustees of the Leland Stanford Junior University**, Stanford, Calif.

[21] Appl. No.: **839,939**

[22] Filed: **Oct. 6, 1977**

[51] **Int. Cl.**² **H04L 9/04**
[52] **U.S. Cl.** **178/22**; 364/900
[58] **Field of Search** 178/22

[56] **References Cited**

PUBLICATIONS

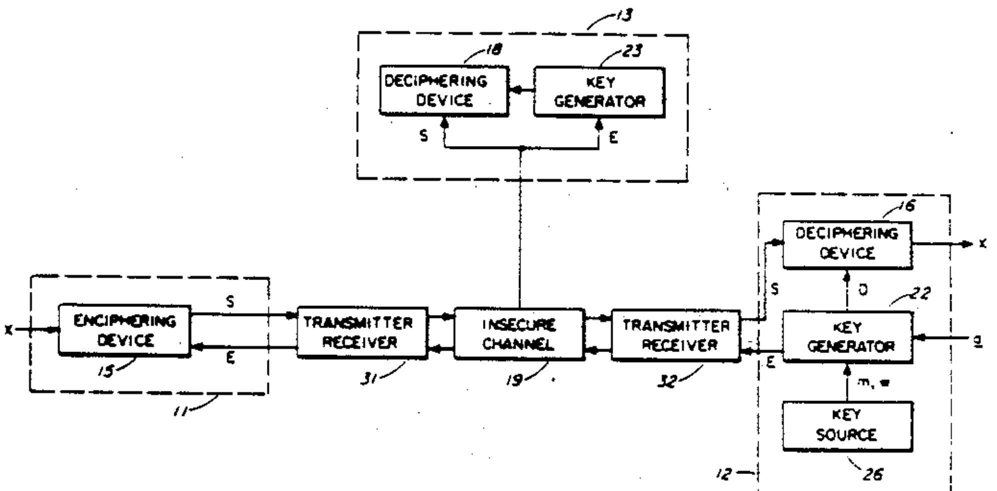
"New Directions in Cryptography," Diffie et al., *IEEE Transactions on Information Theory*, vol. IT22, No. 6, Nov. 1976, pp. 644–654.
"A User Authentication Scheme not Requiring Secrecy in the Computer," Evans, Jr., et al., *Communications of the ACM*, Aug. 1974, vol. 17, No. 8, pp. 437–442.
"A High Security Log-In Procedure," Purdy, *Communications of the ACM*, Aug. 1974, vol. 17, No. 8, pp. 442–445.
Diffie et al., "Multi-User Cryptographic Techniques," *AFIPS Conference Proceedings*, vol. 45, pp. 109–112, Jun. 8, 1976.

Primary Examiner—Howard A. Birmiel

[57] **ABSTRACT**

A cryptographic system transmits a computationally secure cryptogram that is generated from a publicly known transformation of the message sent by the transmitter; the cryptogram is again transformed by the authorized receiver using a secret reciprocal transformation to reproduce the message sent. The authorized receiver's transformation is known only by the authorized receiver and is used to generate the transmitter's transformation that is made publicly known. The publicly known transformation uses operations that are easily performed but extremely difficult to invert. It is infeasible for an unauthorized receiver to invert the publicly known transformation or duplicate the authorized receiver's secret transformation to obtain the message sent.

17 Claims, 13 Drawing Figures



P018930

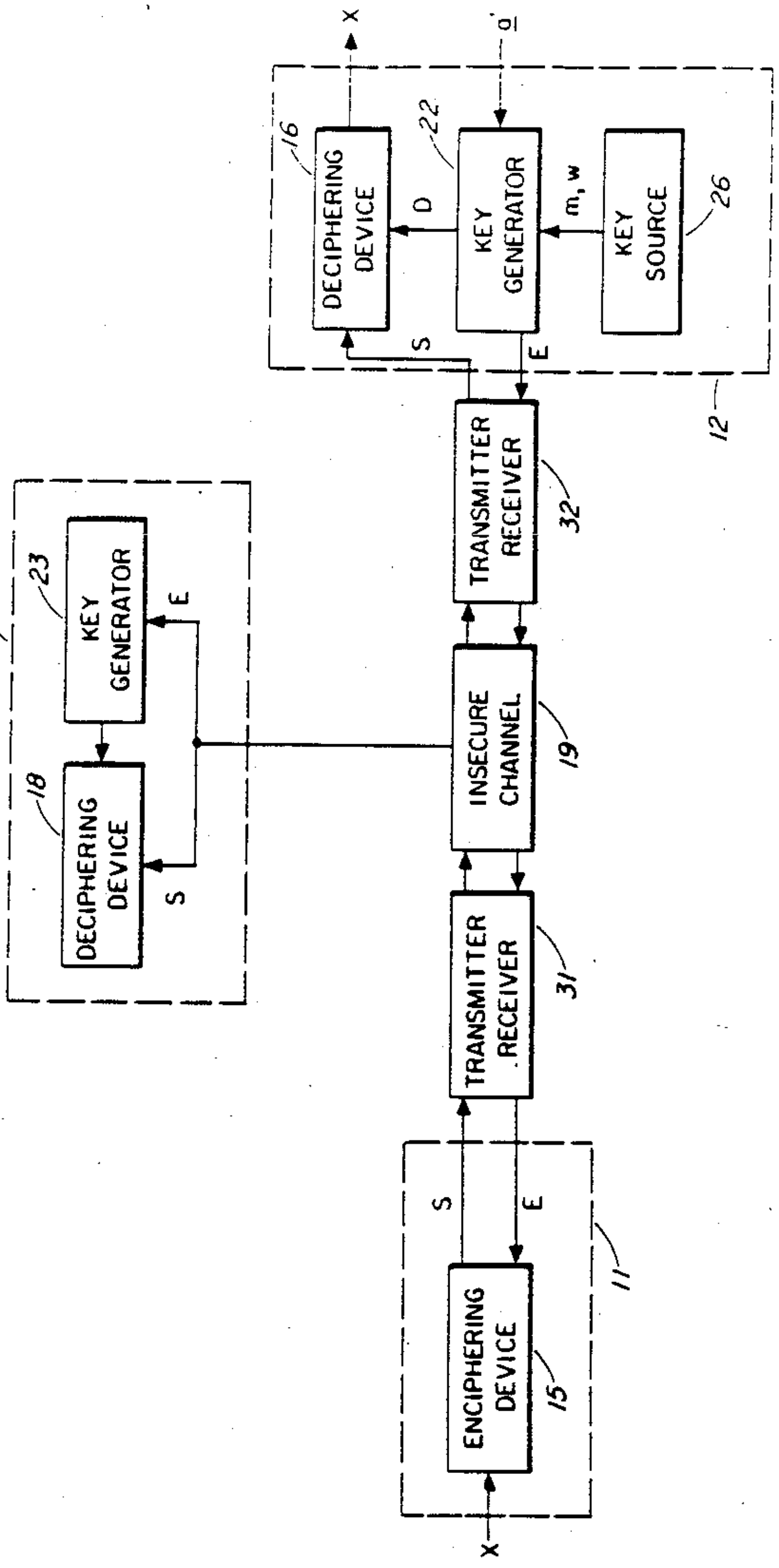
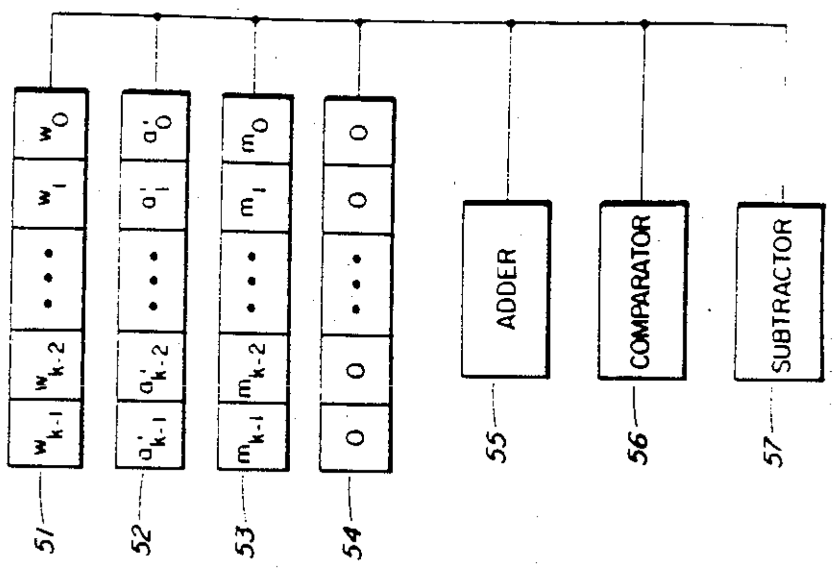
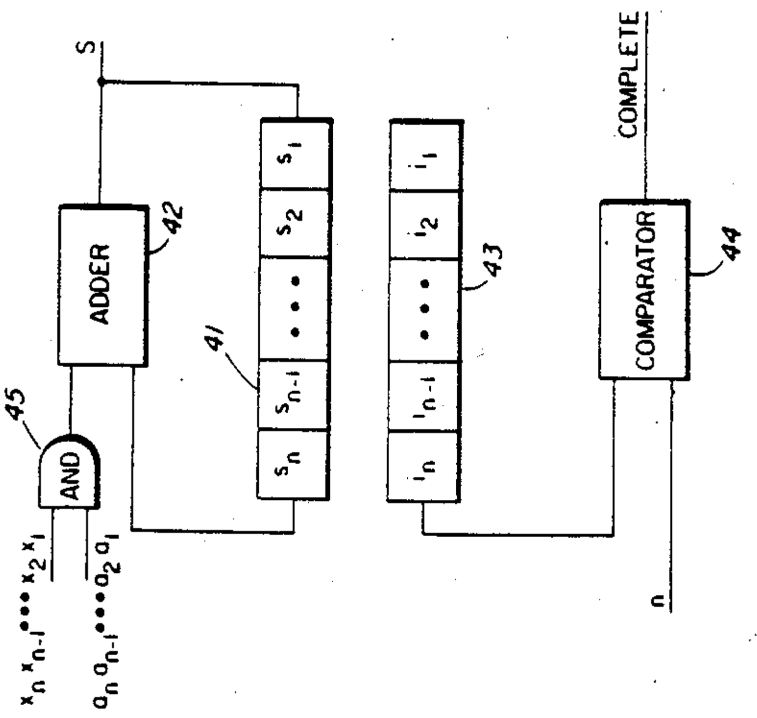

X
ENCIPHERING DEVICE
15
11
S
E
TRANSMITTER RECEIVER
31
INSECURE CHANNEL
19
TRANSMITTER RECEIVER
32
12
DECIPHERING DEVICE
16
KEY GENERATOR
22
KEY SOURCE
26
D
m, w
a
X
13
KEY GENERATOR
23
DECIPHERING DEVICE
18

FIG. 1

FIG. 3



w_{k-1} w_{k-2} • • • w_1 w_0
a'_{k-1} a'_{k-2} • • • a'_1 a'_0
m_{k-1} m_{k-2} • • • m_1 m_0
0 0 • • • 0 0
ADDER
COMPARATOR
SUBTRACTOR
51
52
53
54
55
56
57



S
ADDER
AND
$x_n\, x_{n-1} \cdots x_2\, x_1$
$a_n\, a_{n-1} \cdots a_2\, a_1$
s_n s_{n-1} • • • s_2 s_1
i_n i_{n-1} • • • i_2 i_1
COMPARATOR
COMPLETE
n
41
42
43
44
45

FIG. 2

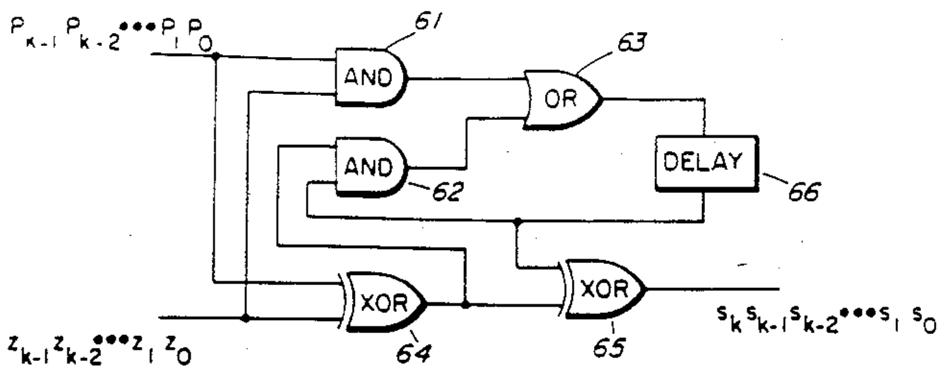

$P_{k-1}\,P_{k-2}\cdots P_1\,P_0$
61
AND
63
OR
AND
62
DELAY
66
XOR
XOR
$z_{k-1}\,z_{k-2}\cdots z_1\,z_0$
64
65
$s_k\,s_{k-1}\,s_{k-2}\cdots s_1\,s_0$

FIG. 4

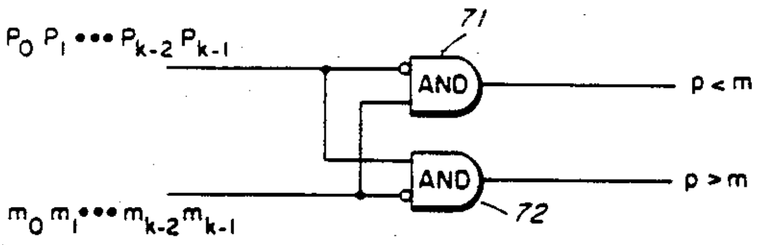

$P_0\,P_1\cdots P_{k-2}\,P_{k-1}$
71
AND
$p < m$
AND
$p > m$
72
$m_0\,m_1\cdots m_{k-2}\,m_{k-1}$

FIG. 5

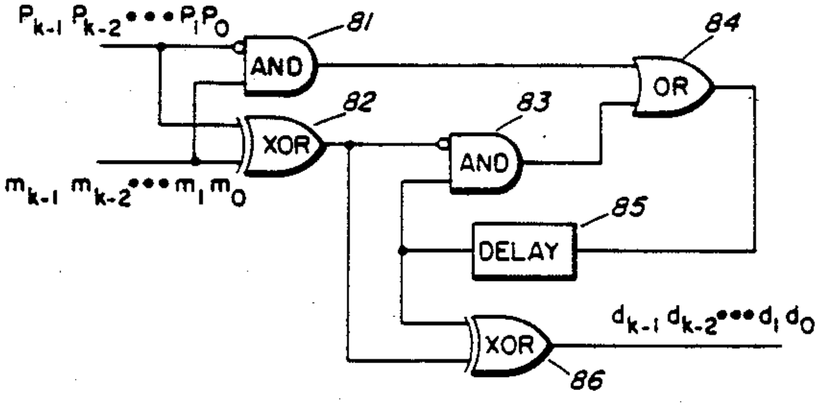
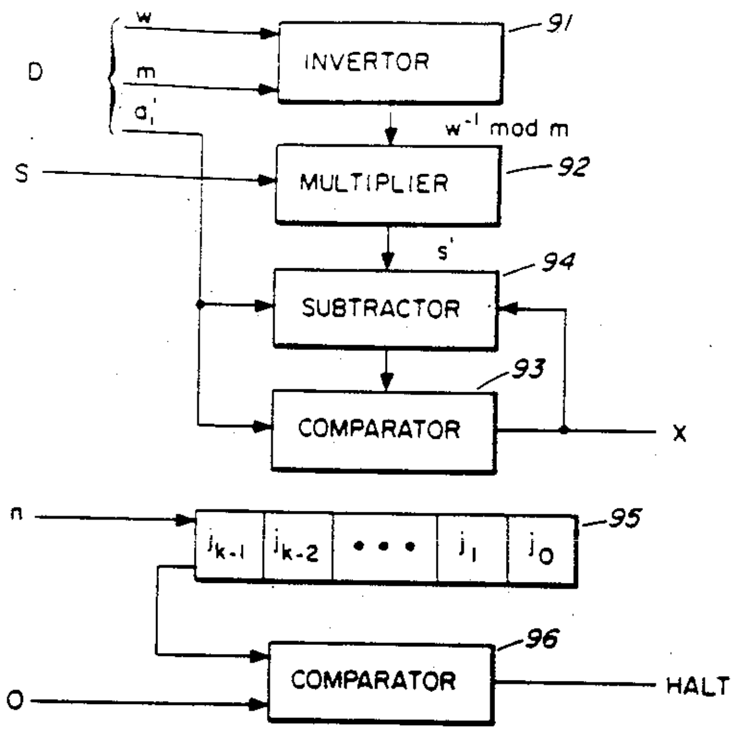
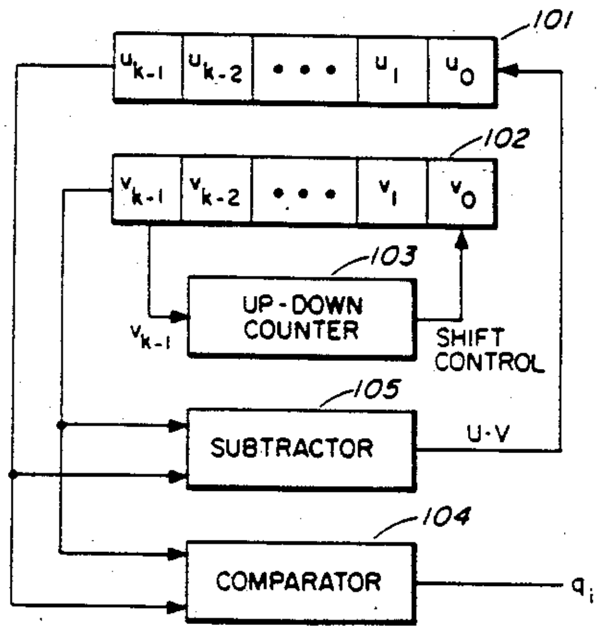

$P_{k-1}\,P_{k-2}\cdots P_1\,P_0$
81
AND
84
OR
82
83
XOR
AND
$m_{k-1}\,m_{k-2}\cdots m_1\,m_0$
85
DELAY
$d_{k-1}\,d_{k-2}\cdots d_1\,d_0$
XOR
86

FIG. 6



D
w
m
a_i'
S
INVERTOR
91
w^{-1} mod m
MULTIPLIER
92
s'
SUBTRACTOR
94
93
COMPARATOR
X
n
j_{k-1}
j_{k-2}
• • •
j_1
j_0
95
96
O
COMPARATOR
HALT

FIG. 7



101
u_{k-1}
u_{k-2}
• • •
u_1
u_0
102
v_{k-1}
v_{k-2}
• • •
v_1
v_0
103
v_{k-1}
UP-DOWN COUNTER
SHIFT CONTROL
105
SUBTRACTOR
U-V
104
COMPARATOR
q_i

FIG. 8

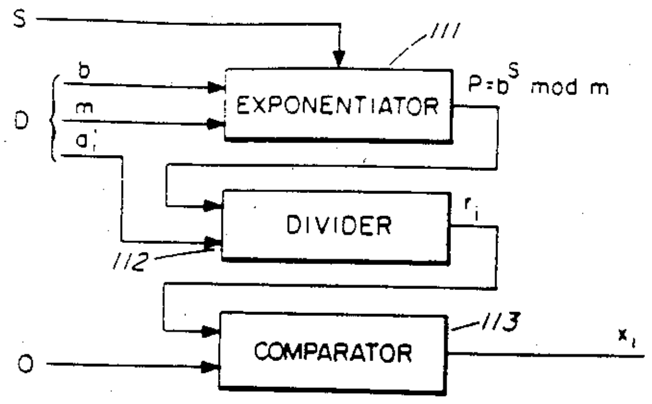

S
111
b
m
D
a_i
EXPONENTIATOR
$P = b^S \bmod m$
DIVIDER
r_i
112
COMPARATOR
113
x_i
0

FIG. 9

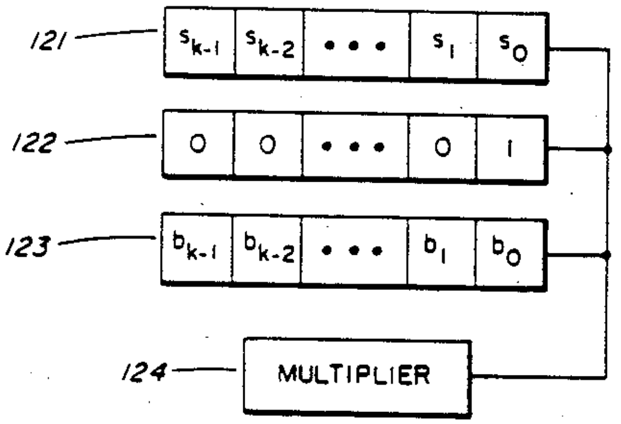

121
s_{k-1}
s_{k-2}
• • •
s_1
s_0
122
0
0
• • •
0
1
123
b_{k-1}
b_{k-2}
• • •
b_1
b_0
124
MULTIPLIER

FIG. 10

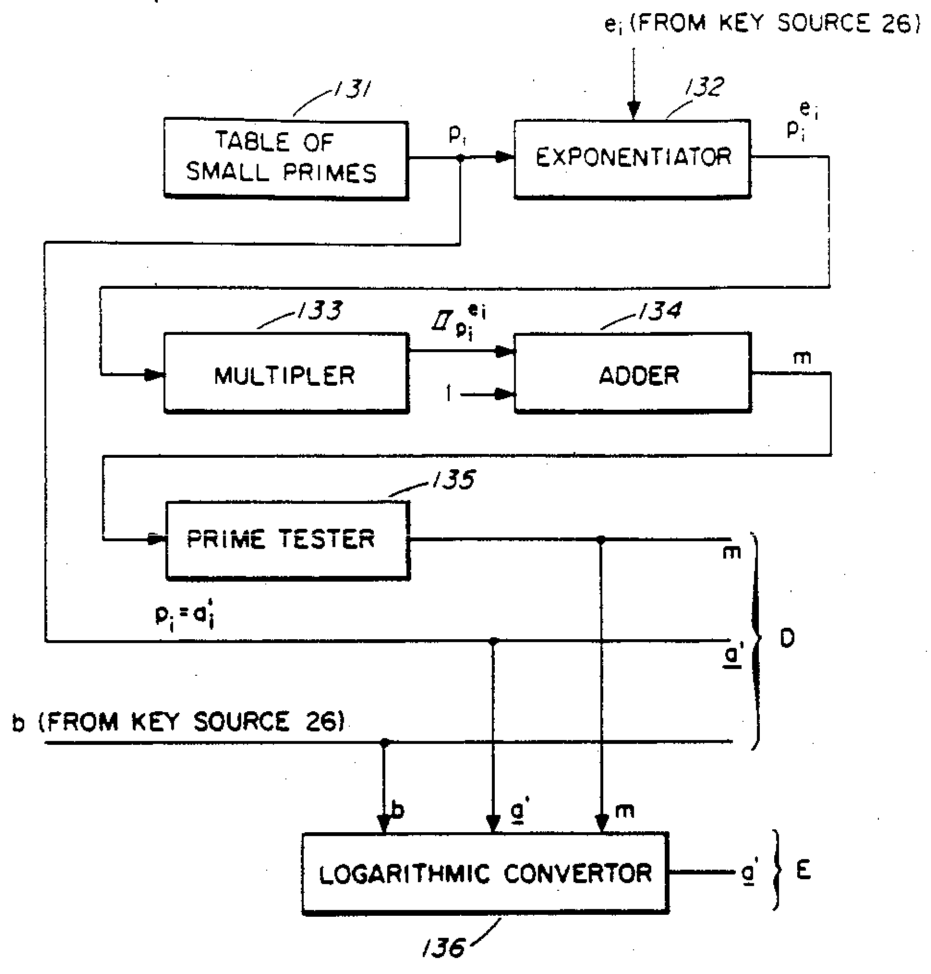

e_i (FROM KEY SOURCE 26)
131
TABLE OF SMALL PRIMES
p_i
132
EXPONENTIATOR
$p_i^{e_i}$
133
MULTIPLER
$\Pi p_i^{e_i}$
134
ADDER
1
m
135
PRIME TESTER
m
$p_i = q_i'$
$\underline{q}'$
D
b (FROM KEY SOURCE 26)
b
$\underline{q}'$
m
LOGARITHMIC CONVERTOR
$\underline{q}'$
E
136

FIG. 11

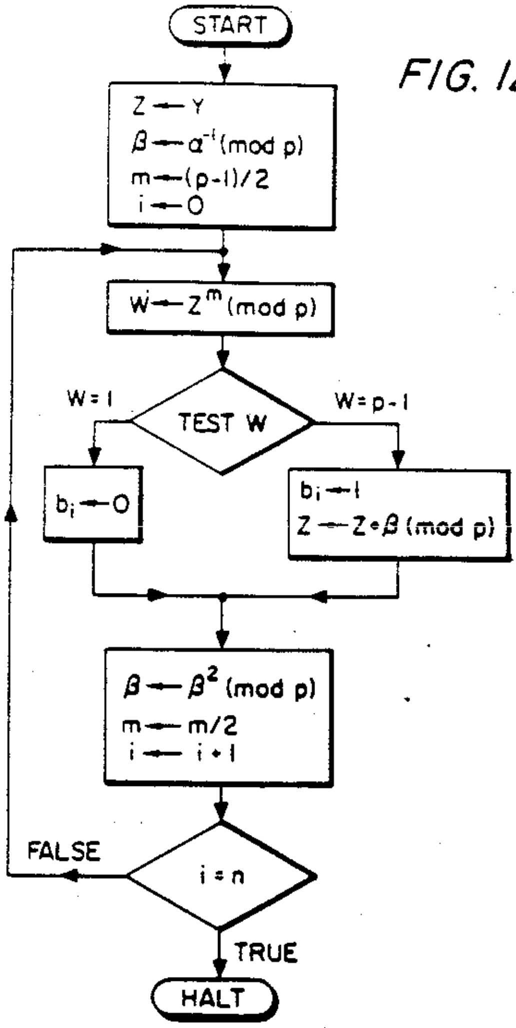

START
$Z \leftarrow Y$
$\beta \leftarrow \alpha^{-1} \pmod p$
$m \leftarrow (p-1)/2$
$i \leftarrow 0$
$W \leftarrow Z^m \pmod p$
TEST W
W=1
W=p-1
$b_i \leftarrow 0$
$b_i \leftarrow 1$
$Z \leftarrow Z \cdot \beta \pmod p$
$\beta \leftarrow \beta^2 \pmod p$
$m \leftarrow m/2$
$i \leftarrow i+1$
i=n
FALSE
TRUE
HALT

FIG. 12

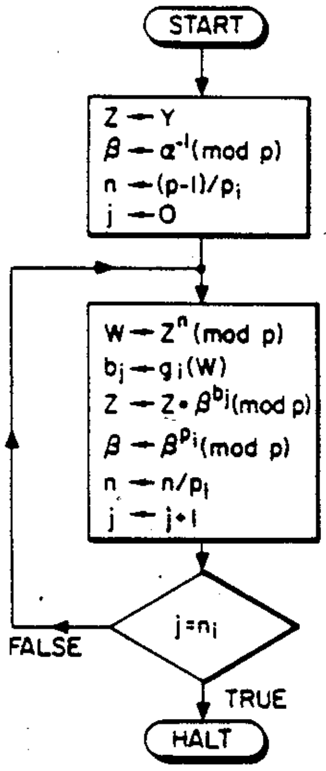

START
$Z \leftarrow Y$
$\beta \leftarrow \alpha^{-1} \pmod p$
$n \leftarrow (p-1)/p_i$
$j \leftarrow 0$
$W \leftarrow Z^n \pmod p$
$b_j \leftarrow g_i(W)$
$Z \leftarrow Z \cdot \beta^{b_j} \pmod p$
$\beta \leftarrow \beta^{p_i} \pmod p$
$n \leftarrow n/p_i$
$j \leftarrow j+1$
$j = n_i$
FALSE
TRUE
HALT

FIG. 13

PUBLIC KEY CRYPTOGRAPHIC APPARATUS AND METHOD

The Government has rights in this invention pursuant to Grant No. ENG-10173 of the National Science Foundation and IPA No. 0005.

BACKGROUND OF THE INVENTION

1. Field of Invention

The invention relates to cryptographic systems.

2. Description of Prior Art

Cryptographic systems are widely used to ensure the privacy and authenticity of messages communicated over insecure channels. A privacy system prevents the extraction of information by unauthorized parties from messages transmitted over an insecure channel, thus assuring the sender of a message that it is being read only by the intended receiver. An authentication system prevents the unauthorized injection of messages into an insecure channel, assuring the receiver of the message of the legitimacy of its sender.

Currently, most message authentication consists of appending an authenticator pattern, known only to the transmitter and intended receiver, to each message and encrypting the combination. This protects against an eavesdropper being able to forge new, properly authenticated messages unless he has also stolen the cipher key being used. However, there is little protection against the threat of dispute; that is, the transmitter may transmit a properly authenticated message, later deny this action, and falsely blame the receiver for taking unauthorized action. Or, conversely, the receiver may take unauthorized action, forge a message to itself, and falsely blame the transmitter for these actions. The threat of dispute arises out of the absence of a suitable receipt mechanism that could prove a particular message was sent to a receiver by a particular transmitter.

One of the principal difficulties with existing cryptographic systems is the need for the sender and receiver to exchange a cipher key over a secure channel to which the unauthorized party does not have access. The exchange of a cipher key frequently is done by sending the key in advance over a secure channel such as private courier or registered mail; such secure channels are usually slow and expensive.

Diffie, et al, in "Multiuser Cryptographic Techniques," *AFIPS-Conference Proceedings*, Vol. 45, pp. 109-112, June 8, 1976, propose the concept of a public key cryptosystem that would eliminate the need for a secure channel by making the sender's keying information public. It is also proposed how such a public key cryptosystem could allow an authentication system which generates an unforgeable message dependent digital signature. Diffie presents the idea of using a pair of keys E and D, for enciphering and deciphering a message, such that E is public information while D is kept secret by the intended receiver. Further, although D is determined by E, it is infeasible to compute D from E. Diffie suggests the plausibility of designing such a public key cryptosystem that would allow a user to encipher a message and send it to the intended receiver, but only the intended receiver could decipher it. While suggesting the plausibility of designing such systems, Diffie presents neither proof that public key cryptosystems exist, nor a demonstration system.

Diffie suggests three plausibility arguments for the existence of a public key cryptosystem: a matrix approach, a machine language approach and a logic mapping approach. While the matrix approach can be designed with matrices that require a demonstrably infeasible cryptanalytic time (i.e., computing D from E) using known methods, the matrix approach exhibits a lack of practical utility because of the enormous dimensions of the required matrices. The machine language approach and logic mapping approach are also suggested, but there is no way shown to design them in such a manner that they would require demonstrably infeasible cryptanalytic time.

Diffie also introduces a procedure using the proposed public key cryptosystems, that could allow the receiver to easily verify the authenticity of a message, but which prevents him from generating apparently authenticated messages. Diffie describes a protocol to be followed to obtain authentication with the proposed public key cryptosystem. However, the authentication procedure relies on the existence of a public key cryptosystem which Diffie did not provide.

SUMMARY AND OBJECTS OF THE INVENTION

Accordingly, it is an object of the invention to allow authorized parties to a conversation (conversers) to converse privately even though an unauthorized party (eavesdropper) intercepts all of their communications.

Another object of this invention is to allow a converser on an insecure channel to authenticate another converser's identity.

Another object of this invention is to provide a receipt to a receiver on an insecure channel to prove that a particular message was sent to the receiver by a particular transmitter. The object being to allow the receiver to easily verify the authenticity of a message, but to prevent the receiver from generating apparently authenticated messages.

An illustrated embodiment of the present invention describes a method and apparatus for communicating securely over an insecure channel, by communicating a computationally secure cryptogram that is a publicly known transformation of the message sent by the transmitter. The illustrated embodiment differs from prior approaches to a public key cryptosystem, as described in "Multiuser Cryptographic Techniques," in that it is both practical to implement and is demonstrably infeasible to invert using known methods.

In the present invention, a receiver generates a secret deciphering key and a public enciphering key, such that the secret deciphering key is difficult to generate from the public enciphering key. The transmitter enciphers a message to be communicated by transforming the message with the public enciphering key, wherein the transformation used to encipher the message is easy to effect but difficult to invert without the secret deciphering key. The enciphered message is then communicated from the transmitter to the receiver. The receiver deciphers the enciphered message by inverting the enciphering transformation with the secret deciphering key.

Another illustrated embodiment of the present invention describes a method and apparatus for allowing a transmitter to authenticate an authorized receiver's identity. The authorized receiver generates a secret deciphering key and a public enciphering key, such that the secret deciphering key is difficult to generate from the public enciphering key. The transmitter enciphers a message to be communicated by transforming the message with the public enciphering key, wherein the transformation used to encipher the message is easy to effect but difficult to invert without the secret deciphering key. The enciphered message is then transmitted from the transmitter to the receiver. The receiver deciphers the enciphered message by inverting the enciphering transformation with the secret deciphering key. The receiver's identity is authenticated to the transmitter by the receiver's ability to decipher the enciphered message.

Another illustrated embodiment of the present invention describes a method and apparatus for providing a receipt for a communicated message. A transmitter generates a secret key and a public key, such that the secret key is difficult to generate from the public key. The transmitter then generates a receipt by transforming a representation of the communicated message with the secret key, wherein the transformation used to generate the receipt is difficult to effect without the secret key and easy to invert with the public key. The receipt is then communicated from the transmitter to the receiver. The receiver inverts the transformation with the public key to obtain the representation of the communicated message from the receipt and validates the receipt by comparing the similarity of the representation of the communicated message with the communicated message.

Additional objects and features of the present invention will appear from the description that follows wherein the preferred embodiments have been set forth in detail in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram of a public key cryptosystem that transmits a computationally secure cryptogram over an insecure communication channel.

FIG. 2 is a block diagram of an enciphering device for enciphering a message into ciphertext in the public key cryptosystem of FIG. 1.

FIG. 3 is a block diagram of a multiplier for performing modular multiplications in the deciphering device of FIG. 7, the exponentiator of FIG. 10, and the public key generator of FIG. 11.

FIG. 4 is a detailed schematic diagram of an adder for performing additions in the enciphering device of FIG. 2, the multiplier of FIG. 3, and the public key generator of FIG. 11.

FIG. 5 is a detailed schematic diagram of a comparator for performing magnitude comparisons in the enciphering device of FIG. 2, the multiplier of FIG. 3, the deciphering device of FIG. 7, the divider of FIG. 8, and the alternative deciphering device of FIG. 9.

FIG. 6 is a detailed schematic diagram of a subtractor for performing subtraction in the multiplier of FIG. 3, the deciphering device of FIG. 7, and the dividier of FIG. 8.

FIG. 7 is a block diagram of a deciphering device for deciphering a ciphertext into message in the public key cryptosystem of FIG. 1.

FIG. 8 is a block diagram of a divider for performing division in the invertor of FIG. 7 and the alternative deciphering device of FIG. 9.

FIG. 9 is a block diagram of an alternative deciphering device for deciphering a ciphertext into message in the public key cryptosystem of FIG. 1.

FIG. 10 is an exponentiator for raising various numbers to various powers in modulo arithmetic in the alternative deciphering device of FIG. 9 and the public key generator of FIG. 11.

FIG. 11 is a public key generator for generating the public enciphering key in the public key cryptosystem of FIG. 1.

FIG. 12 is a flow chart for the algorithm of the logarithmic converter of FIG. 11 when p−1 is a power of 2.

FIG. 13 is a flow chart for the algorithm for computing the coefficients $\{b_j\}$ of the expansion

$$x(\bmod p_i^{n_i}) = \sum_{j=0}^{n_i-1} b_j p_i^j$$

where $0 \leqq b_j \leqq p_i - 1$, of the logarithmic convertor of FIG. 11, when p−1 is not a power of 2.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Referring to FIG. 1, a public key cryptosystem is shown in which all transmissions take place over an insecure communication channel 19, for example a telephone line. Communication is effected on the insecure channel 19 between transmitter 11 and receiver 12 using transmitter-receiver units 31 and 32, which may be modems such as Bell 201 modems. Transmitter 11 possesses an unenciphered or plaintext message X to be communicated to receiver 12. Transmitter 11 and receiver 12 include an enciphering device 15 and deciphering device 16 respectively, for enciphering and deciphering information under the action of an enciphering key E on line E and a reciprocal deciphering key D on line D. The enciphering and deciphering devices 15 and 16 implement inverse transformations when loaded with the corresponding keys E and D. For example, the keys may be a sequence of random letters or digits. The enciphering device 15 enciphers the plaintext message X into an enciphered message or ciphertext S that is transmitted by transmitter 11 through the insecure channel 19; the ciphertext S is received by receiver 12 and deciphered by deciphering device 16 to obtain the plaintext message X. An unauthorized party or eavesdropper 13 is assumed to have key generator 23 and deciphering device 18 and to have access to the insecure channel 19, so if he knew the deciphering key D he could decipher the ciphertext S to obtain the plaintext message X.

The example system makes use of the difficulty of the so-called "knapsack problem." Definitions of the knapsack problem exist in the literature, for example, Ellis Horowitz and Sartaj Sahni, "Computing Partitions with Applications to the Knapsack Problem", *JACM*, Vol. 21, No. 2, April 1974, pp. 277–292; and O. H. Ibarra and C. E. Kim, "Fast Approximation Algorithms for the Knapsack and Sum of Subset Problems", *JACM*, Vol. 22, No. 4, October 1975, pp. 464–468. The definition used here is adapted from R. M. Karp, "Reducibility Among Combinatorial Problems" in Complexity of Computer Computations, by R. E. Miller and J. W. Thatcher, eds., Plenum Press, New York (1972), pp. 85–104. Simply stated, given a one-dimensional knapsack of length S and a vector a composed of n rods of lengths $a_1, a_2, \ldots a_n$, the knapsack problem is to find a subset of the rods which exactly fills the knapsack, if such a subset exists. Equivalently, find a binary n-vector x of 0's and 1's such that $S = a*x$, if such an x exists. (* applied to vectors denotes dot product, applied to scalars denotes normal multiplication).

A supposed solution, x, is easily checked in at most n additions; but, to the best of current knowledge, finding a solution requires a number of operations which grows exponentially in n. Exhaustive trial and error search over all 2^n possible x's is computationally infeasible if n is larger than one or two hundred. Thus, it is computationally infeasible to invert the transformation; such transformations are characterized by the class of mathematical functions known as one-way cipher functions. A task is considered computationally infeasible if its cost as measured by either the amount of memory used or the computing time is finite but impossibly large, for example, on the order of approximately 10^{30} operations with existing computational methods and equipment.

Theory suggests the difficulty of the knapsack problem because it is an NP-complete problem, and is therefore one of the most difficult computational problems of a cryptographic nature. (See for example, A. V. Aho, J. E. Hopcraft and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Reading, Ma.; Addison-Wesley, 1974, pp. 363–404.) Its degree of difficulty, however, is dependent on the choice of a. If $a=(1, 2, 4, \ldots 2^{(n-1)})$, then solving for x is equivalent to finding the binary representation of S. Somewhat less trivially, if for all i,

$$a_i > \sum_{j=1}^{i-1} a_j \qquad (1)$$

then x is also easily found: $x_n=1$ if and only if $S \geqq a_n$, and, for $i=n-1, n-2, \ldots 1$, $x_i=1$ if and only if

$$S - \sum_{j=i+1}^{n} x_j * a_j \geqq a_i \qquad (2)$$

If the components of x are allowed to take on integer values between 0 and 1 then condition (1) can be replaced by

$$a_i > l \sum_{j=1}^{i-1} a_j$$

and x_i can be recovered as the integer part of

$$(S - \sum_{j=i+1}^{n} x_j * a_j)/a_i.$$

Equation (2) for evaluating x_i when x_i is binary valued is equivalent to this rule for $l=1$.

A trap door knapsack is one in which careful choice of a allows the designer to easily solve for any x, but which prevents anyone else from finding the solution. Two methods will be described for constructing trap door knapsacks, but first a description of their use in a public key cryptosystem as shown in FIG. 1 is provided. Receiver 12 generates a trap door knapsack vector a, and either places it in a public file or transmits it to transmitter 11 over the insecure channel 19. Transmitter 11 represents the plaintext message X as a vector x of n 0's and 1's, computes $S=a*x$, and transmits S to receiver 12 over the insecure channel 19. Receiver 12 can solve S for x, but it is infeasible for eavesdropper 13 to solve S for x.

In one method for generating trap door knapsacks, the key generator 22, uses random numbers generated by key source 26 to select two large integers, m and w, such that w is invertible modulo m, (i.e., so that m and

w have no common factors except 1). For example, the key source 26 may contain a random number generator that is implemented from noisy amplifiers (e.g., Fairchild μ 709 operational amplifiers) with a polarity detector. The key generator 22 is provided a knapsack vector, a' which satisfies (1) and therefore allows solution of $S'=a'*x$, and transforms the easily solved knapsack vector a' into a trap door knapsack vector a via the relation

$$a_i = w * a_i' \bmod m \qquad (3)$$

The vector a serves as the public enciphering key E on line E, and is either placed in a public file or transmitted over the insecure channel 19 to transmitter 11. The enciphering key E is thereby made available to both the transmitter 11 and the eavesdropper 13. The transmitter 11 uses the enciphering key E, equal to a, to generate the ciphertext S from the plaintext message X, represented by vector x, by letting $S=a*x$. However, because the a_i may be psuedo-randomly distributed, the eavesdropper 13 who knows a, but not w or m, cannot feasibly solve a knapsack problem involving a to obtain the desired message x.

The deciphering device 16 of receiver 12 is given w, m and a' as its secret deciphering key D, and can easily compute

$$\begin{aligned} S' &= 1/w * S \bmod m && (4)\\ &= 1/w * \Sigma x_i * a_i \bmod m && (5)\\ &= 1/w * \Sigma x_i * w * a_i' \bmod m && (6)\\ &= \Sigma x_i * a_i' \bmod m && (7) \end{aligned}$$

If m is chosen so that

$$m > \Sigma a_i' \qquad (8)$$

then (7) implies that S' is equal to $\Sigma x_i * a_i'$ in integer arithmetic as well as mod m. This knapsack is easily solved for x, which is also the solution to the more difficult trap door knapsack problem $S=a*x$. Receiver 12 is therefore able to recover the plaintext message X, represented as the binary vector x. But, the eavesdropper 13's trap door knapsack problem can be made computationally infeasible to solve, thereby preventing the eavesdropper 13 from recovering the plaintext message X.

To help make these ideas more clear, an illustrative example is given in which $n=5$. Taking $m=8443$, $a'=(171, 196, 457, 1191, 2410)$, and $w=2550$, then $a=(5457, 1663, 216, 6013, 7439)$. Given $x=(0, 1, 0, 1, 1)$ the enciphering device 15 computes $S=1663+6013+7439=15115$. The deciphering device 16 uses Euclid's algorithm (see for instance, D. Knuth, *The Art of Computer Programming*, vol. II, Addison-Wesley, 1969, Reading Ma.) to compute $1/w=3950$ and then computes

$$\begin{aligned} S' &= 1/w * S \bmod m && (9)\\ &= 3950 * 15115 \bmod 8443 &&\\ &= 3797 && \end{aligned}$$

Because $S' > a_5'$, the deciphering device 16 determines that $x_5=1$. Then, using (2) for the a' vector, it determines that $x_4=1$, $x_3=0$, $x_2=1$, $x_1=0$ or $x=(0, 1, 0, 1, 1)$, which is also the correct solution to $S=a*x$.

The eavesdropper, 13 who does not know m, w or a' has great difficulty in solving for x in $S=a*x$ even though he knows the method used for generating the trap door knapsack vector a. His task can be made infeasible by choosing larger values for n, m, w and a'. His task can be further complicated by scrambling the order of the a_i, and by adding different random multiples of m to each of the a_i.

The example given was extremely small in size and only intended to illustrate the technique. Using $n=100$ (which is the lower end of the usable range for high security systems at present) as a more reasonable value, it is suggested that m be chosen approximately uniformly from the numbers between $2^{201}+1$ and $2^{202}-1$; that a_1' be chosen uniformly from the range $(1, 2^{100})$; that a_2' be chosen uniformly from $(2^{100}+1, 2*2^{100})$; that a_3' be chosen uniformly from $(3\times 2^{100}+1, 4*2^{100})$; . . . and that a_i' be chosen uniformly from $((2^{i-1}-1)*2^{100}+1, 2^{i-1}*2^{100})$; and that w' be chosen uniformly from $(2, m-2)$ and then divided by the greatest common divisor of (w', m) to yield w.

These choices ensure that (8) is met and that an eavesdropper 13 has at least 2^{100} possibilities for each parameter and hence cannot search over them all.

The enciphering device 15 is shown in FIG. 2. The sequence of integers $a_1, a_2, \ldots a_n$ is presented sequentially in synchronization with the sequential presentation of 0's and 1's of $x_1, x_2, \ldots x_n$. The S register 41 is initially set to zero. If $x_i=1$ the S register 41 contents are a_i are added by adder 42 and the result placed in the S register 41. If $x_i=0$ the contents of the S register 41 are left unchanged. In either event, i is replaced by i+1 until i=n, in which case the enciphering operation is complete. The i register 43 is initially set to zero and incremented by 1 after each cycle of the enciphering device. Either the adder 42, or a special up counter can be used to increment the i register 43 contents. With the range of values suggested above, the S and i registers 41 and 43 both can be obtained from a single 1024 bit random access memory (RAM) such as the Intel 2102. The implementation of the adder 42 will be described in more detail later, as will the implementation of a comparator 44 required for comparing i and n to determine when the enciphering operation is complete.

The key generaor 22 comprises a modulo m multiplier, such as that depicted in FIG. 3, for producing $a_i=w*a_i'$ mod m. The two numbers w and a_i' to be multiplied are loaded into the W and A' registers 51 and 52 respectively, and m is loaded into the M register 53. The product $w*a_i'$ modulo m will be produced in the P register 54 which is initially set to zero. If k, the number of bits in the binary representation of m, is 200, then all four registers can be obtained from a single 1024 bit RAM such as the Intel 2102. The implementation of FIG. 3 is based on the fact that wa_i' mod m $= w_oa_i'$ mod m$+2\ w_1a_i'$ mod m$+4\ w_2a_i'$ mod m$+\ \ldots\ +2^{k-1}\ w_{k-1}a_i'$ mod m.

To multiply w times a_i', if the rightmost bit, containing w_o of the W register 51 is 1 then the contents of the A' register 53 are added to the P register 54 by adder 55. If $w_o=0$, then the P register 54 is unchanged. Then the M and P register contents are compared by comparator 56 to determine if the contents of the P register 54 are greater than or equal to m, the contents of the M register 53. If the contents of the P register 54 are greater than or equal to m then subtractor 57 subtracts m from the contents of the P register 54 and places the difference in the P register 54, if less than m the P register 54 is unchanged.

Next, the contents of W register 51 are shifted one bit to the right and a 0 is shifted in at the left so its contents become $0w_{k-1}\ w_{k-2}\ldots w_2w_1$, so that w is ready for computing $2w_1a'$ mod m. The quantity of 2a mod m is computed for this purpose by using adder 55 to add a to itself, using comparator 56 to determine if the result, 2a, is less than m, and using subtractor 57 for subtracting m from 2a' if the result is not less than m. The result, 2a' mod m is then stored in the A' register 52. The rightmost bit, containing w_1, of the W register 51 is then examined, as before, and the process repeats.

This process is repeated a maximum of k times or until the W register 51 contains all 0's, at which point wa' modulo m is stored in the P register 54.

As an example of these operations, consider the problem of computing 7×7 modulo 23. The following steps show the successive contents of the W, A' and P registers which result in the answer $7\times 7=3$ modulo 23.

i	W (in binary)	A'	P
0	00111	7	0
1	00011	14	0 + 7 = 7
2	00001	5	7 + 14 = 21
3	00000	10	21 + 5 = 3 mod 23

FIG. 4 depicts an implementation of an adder 42 or 55 for adding two k bit numbers p and z. The numbers are presented one bit at a time to the device, low order bit first, and the delay element is initially set to 0. (The delay represents the binary carry bit.) The AND gate 61 determines if the carry bit should be a one based on p_i and z_i both being 1 and the AND gate 62 determines if the carry should be 1 based on the previous carry being a 1 and one of p_i or z_i being 1. If either of these two conditions is met, the OR gate 63 has an output of 1 indicating a carry to the next stage. The two exclusive-or (XOR) gates 64 and 65 determine the i^{th} bit of the sum, s_i, as the modulo-2 sum of p_i, z_i and the carry bit from the previous stage. The delay 66 stores the previous carry bit. Typical parts for implementing these gates and the delay are SN7400, SN7404, and SN7474.

FIG. 5 depicts an implementation of a comparator 44 or 56 for comparing two numbers p and m. The two numbers are presented one bit at a time, high order bit first. If neither the p<m nor the p>m outputs have been triggered after the last bits p_o and m_o have been presented, then $p=m$. The first triggering of either the p<m or the p>m output causes the comparison operation to cease. The two AND gates 71 and 72 each have one input inverted (denoted by a circle at the input). An SN7400 and SN7404 provide all of the needed logic circuits.

FIG. 6 depicts an implementation of a subtractor 57 for subtracting two numbers. Because the numbers subtracted in FIG. 3 always produce a non-negative difference, there is no need to worry about negative differences. The larger number, the minuend, is labelled p and the smaller number, the subtrahend, is labelled m. Both p and m are presented serially to the subtractor 57, low order bit first. AND gates 81 and 83, OR gate 84 and XOR gate 82 determine if borrowing (negative carrying) is in effect. A borrow occurs if either $p_i=0$ and $m_i=1$, or $p_i=m_i$ and borrowing occurred in the previous stage. The delay 85 stores the previous borrow state. The ith bit of the difference, d_i, is computed as the XOR, or modulo-2 difference, of p_i, m_i and the borrow bit. The output of XOR gate 82 gives the modulo-2 difference between p_i and m_i, and XOR gate 86 takes the modulo-2 difference of this with the previous borrow bit. Typical parts for implementing these gates and the delay are SN7400, SN7404 and SN7474.

The deciphering device 16 is depicted in FIG. 7. It is given the ciphertext S, and the deciphering key consisting of w, m and a', and must compute x.

To compute x, first, w and m are input to a modulo m invertor 91 which computes w^{-1} mod m. It then uses the modulo m multiplier 92 to compute $S'=w^{-1}S$ mod m. As noted in equations (7) and (8), $S'=a'*x$, which is easily solved for x. The comparator 93 then compares S' with a_n' and decides that $x_n=1$ if $S' \geqq a_n'$ and that $x_n=0$ if $S'<a_n'$. If $x_n=1$, S' is replaced by $S'-a_n'$, computed by the subtractor 94. If $x_n=0$, S' is unchanged. The process is repeated for a_{n-1}' and x_{n-1} and continues until x is computed. The j register 95 is initially set to n and is decremented by 1 after each stage of the deciphering process until $j=0$ results, causing a halt to the process and signifying x is computed. Either the subtractor 94 or a down counter can be used to decrement the contents of the j register 95. The comparator 96 can be used to compare the contents of the j register 95 with zero to determine when to halt the process. The modulo m multiplier 92 is detailed in FIG. 3; the comparator 93 is detailed in FIG. 5; and, the subtractor 94 is detailed in FIG. 6. The modulo m invertor 91 can be based on a well known extended version of Euclid's algorithm. (See for instance, D. Knuth, *The Art of Computer Programming*, Vol. II, Addison-Wesley, 1969, Reading, Ma., p. 302 and p. 315 problem 15.) As described by Knuth, an implementation requires six registers, a comparator, a divider and a subtractor. All of these devices have already been detailed with the exception of the divider.

FIG. 8 details an apparatus for dividing an integer u by another integer v to compute a quotient q and a remainder r, such that $0 \leqq r \leqq v-1$. First, u and v, represented as binary numbers, are loaded into the U and V registers 101 and 102, respectively. Then v, the contents of the V register 102, are shifted to the left until a 1 appears in v_{k-1}, the leftmost bit of the V register 102. This process can be effected by using the complement of v_{k-1} to drive the shift control on a shift register, such as the Signetics 2533, which was initially set to zero. The contents of the up-down counter 103 equal the number of bits in the quotient less one.

After this initialization, v, the contents of the V register 102 are compared with the contents of the U register 101 by the comparator 104. If $v>u$ then q_n, the most significant bit of the quotient, is 0 and u is left unchanged. If $v \leqq u$ then $q_n=1$ and u is replaced by u-v as computed by the subtractor 105. In either event, v is shifted to the right one bit and the $v>u$ comparison is repeated to compute q_{n-1}, the next bit in the quotient.

This process is repeated, with the up-down counter 103 being decremented by 1 after each iteration until it contains zero. At that point, the quotient is complete and the remainder r is in the U register 101.

As an example, consider dividing 14 by 4 to produce $q=3$ and $r=2$ with $k=4$ being the register size. Because $u=14=1110$ and $v=4=0100$ in binary form, the V register 101 is left shifted only once to produce $v=1000$. After this initialization, it is found that $v \leqq u$ so the first quotient bit $q_1=1$, and u is replaced by u-v; v is replaced by v right shifted one bit and the up-down counter 103 is decremented to zero. This signals that the last quotient bit, q_0, is being computed, and that after the present iteration the remainder, r, is in the U register. The following sequence of register contents helps in following these operations.

U	V	Counter
1110	1000	[illegible]
0110	0100	0
0010	—	halt

It is seen that $q=11$ in binary form and is equivalent to $q=3$, and that $r=0010$ in binary form and is equivalent to $r=2$.

Another method for generating a trap door knapsack vector a uses the fact that a multiplicative knapsack is easily solved if the vector entries are relatively prime. Given $a'=(6, 11, 35, 43, 169)$ and a partial product $P=2838$, it is easily determined that $P=6*11*43$ because 6, 11 and 43 evenly divide P but 35 and 169 do not. A multiplicative knapsack is transformed into an additive knapsack by taking logarithms. To make both vectors have reasonable values, the logarithms are taken over GF(m), the Galois (finite) field with m elements, where m is a prime number. It is also possible to use non-prime values of m, but the operations are somewhat more difficult.

A small example is again helpful. Taking $n=4$, $m=257$, $a'=(2, 3, 5, 7)$ and the base of the logarithms to be $b=131$ results in $a=(80, 183, 81, 195)$. That is $131^{80}=2$ mod. 257; $131^{183}=3$ mod 257; etc. Finding logarithms over GF (m) is relatively easy if $m-1$ has only small prime factors.

Now, if the deciphering device 16 is given $S=183+81=264$, it uses the deciphering key D consisting of m, a' and b, to compute

$$\begin{aligned} S' &= b^S \bmod m & (10)\\ &= 131^{264} \bmod 257 \\ &= 15 \\ &= 3*5 \\ &= a_1'^0 * a_2'^1 * a_3'^1 a_4'^0 \end{aligned}$$

which implies that $x=(0, 1, 1, 0)$. This is because

$$\begin{aligned} b^S &= b^{(\Sigma a_i * x_i)} & (11)\\ &= \pi\, b^{(a_i * x_i)} & (12)\\ &= \pi\, a_i'^{x_i} \bmod m & (13) \end{aligned}$$

However, it is necessary that

$$\prod_{i=1}^{n} a_i' < m \quad (14)$$

to ensure that $\pi a_i'^{x_i}$ mod m equals $\pi a_i'^{x_i}$ in arithmetic over the integers.

The eavesdropper 13 knows the enciphering key E, comprised of the vector a, but does not know the deciphering key D and faces a computationally infeasible problem.

The example given was again small and only intended to illustrate the technique. Taking $n=100$, if each a_i' is a random, 100 bit prime number, then m would have to be approximately 10,000 bits long to ensure that (14) is met. While a 100:1 data expansion is acceptable in certain applications (e.g., secure key distribution over an insecure channel), it probably is not necessary for an opponent to be so uncertain of the a_i'. It is even possible to use the first n primes for the a_i', in which case m could be as small as 730 bits long when $n=100$ and still meet condition (14). As a result, there is a possible tradeoff between security and data expansion.

In this embodiment, the enciphering device 15 is of the same form as detailed in FIG. 2 and described above. The deciphering device 16 of the second embodiment is detailed in FIG. 9. The ciphertext S and part of the deciphering key D, namely b and m, are used by the exponentiator 111 to compute $P=b^S$ mod m. As noted in equations (12) to (14) and in the example, P is a partial product of the $\{a_i\}$, also part of the deciphering key D. The divider 112 divides P by a_i' for $i=1, 2, \ldots n$ and delivers only the remainder r_i, to the comparator 113. If $r_i=0$ then a_i' evenly divides P and $x_i=1$. If $r_i \neq 0$ then $x_i=0$. The divider 112 may be implemented as detailed in FIG. 8 and described above. The comparator 113 may be implemented as detailed in FIG. 5 and described above; although, more efficient devices exist for comparing with zero.

The exponentiator 111, for raising b to the S power modulo m, can be implemented in electronic circuitry as shown in FIG. 10. FIG. 10 shows the initial contents of three registers 121, 122 and 123. The binary representation of S ($s_{k-1}s_{k-2} \ldots s_1s_0$) is loaded into the S register 121; 1 is loaded into the R register 122; and the binary representation of b is loaded into the B register 123, corresponding to $i=0$. The number of bits k in each register is the least integer such that $2^k \geqq m$. If $k=200$, then all three registers can be obtained from a single 1024 bit random access memory (RAM) such as the Intel 2102. The implementation of multiplier 124, for multiplying two numbers modulo m, has been described in detail in FIG. 3.

Referring to FIG. 10, if the low order bit, containing s_0, of the S register 121 equals 1 then the R register 122 and the B register 123 contents are multiplied modulo m and their product, also a k bit quantity, replaces the contents of the R register 122. If $s_0=0$, the R register 122 contents are left unchanged. In either case, the B register 123 is then loaded twice into the multiplier 124 so that the square, modulo m, of the B register 123 contents is computed. This valve, $b^{(2^i+1)}$, replaces the contents of the B register 123. The S register 121 contents are shifted one bit to the right and a 0 is shifted in at the left so its contents are now $0s_{k-1}s_{k-2} \ldots s_2s_1$.

The low order bit, containing s_1, of the S register 121 is examined. If it equals 1 then, as before, the R register 122 and B register 123 contents are multiplied modulo m and their product replaces the contents of the R register 122. If the low order bit equals 0 then the R register 122 contents are left unchanged. In either case, the contents of the B register 123 are replaced by the square, modulo m, of the previous contents. The S register 121 contents are shifted one bit to the right and a 0 is shifted in at the left so its contents are now $00s_{k-1}s_{k-2} \ldots s_3s_2$.

This process continues until the S register 121 contains all 0's, at which point the value of b^S modulo m is stored in the R register 122.

An example is helpful in following this process. Taking $m=23$, we find $k=5$ from $2^k \geqq m$. If $b=7$ and $S=18$ then

$$b^s=7^{18}=1628413597910449=23(70800591213497)+18$$

so b^s modulo m equals 18. This straightforward but laborious method of computing b^s modulo m is used as a check to show that the method of FIG. 10, shown below, yields the correct result. The R register 122 and B register 123 contents are shown in decimal form to facilitate understanding.

i	S (in binary)	R	B
0	10010	1	7
1	01001	1	3
2	00100	3	9
3	00010	3	12
4	00001	3	6
5	00000	18	13

The row marked $i=0$ corresponds to the initial contents of each register, $S=18$, $R=1$ and $B=b=7$. Then, as described above, because the right most bit of S register 121 is 0, the R register 122 contents are left unchanged, the contents of the B register 123 are replaced by the square, modulo 23, of its previous contents ($7^2=49=2\times23+3=3$ modulo 23), the contents of the S register 121 are shifted one bit to the right, and the process continues. Only when $i=1$ and 4 do the rightmost bit of the S register 121 contents equal 1, so only going from $i=1$ to 2 and from $i=4$ to 5 is the R register 122 replaced by RB modulo m. When $i=5$, $S=0$ so the process is complete and the result, 18, is in the R register 122.

Note that the same result, 18, is obtained here as in the straightforward calculation of 7^{18} modulo 23, but that here large numbers never resulted.

Another way to understand the process is to note that the B register 123 contains b, b^2, b^4, b^8 and b^{16} when $i=0$, 1, 2, 3 and 4 respectively, and that $b^{18}=b^{16}b^2$, so only these two values need to be multiplied.

The key generator 22 used in the second embodiment is detailed in FIG. 11. A table of n small prime numbers, p_i, is created and stored in source 131, which may be a read only memory such as the Intel 2316E. The key source 26, as described above, generates random numbers, e_i. The small prime numbers from the source 131 are each raised to a different power, represented by a random number e_i from key source 26, by the exponentiator 132 to generate $p_i^{e_i}$ for $i=1$ to n. The multiplier 133 then computes the product of all the $p_i^{e_i}$ which may be represented as

$$\prod^{n} p_i^{e_i}$$

The product of all the

$$p_1^{e_1} \cdot \prod^{n} p_i^{e_i} .$$

then is incremented by one by adder 134 to generate the potential value of m. If it is desired that m be prime, the potential value of m may be tested for primeness by prime tester 135.

Prime testers for testing a number m for primeness when the factorization of m−1 is known

$$\text{(as here, } m-1=\prod_{i=1}^{n} p_i^{e_i}\text{),}$$

are well documented in the literature. (See for instance, D. Knuth, *The Art of Computer Programming*, vol. II, Seminumerical Algorithms, pp. 347–48.) As described in the above reference, the prime tester 135 requires only a means for exponentiating various numbers to various powers modulo m, as described in FIG. 10. When a potential value of m is found to be prime, it is output by the public key generator of FIG. 11 as the variable m. The a' vector's elements, a_i', can then be chosen to be the n small prime numbers, p_i, from source 131.

The base, b, of the logarithms is then selected as a random number by the key source 26.

The elements of the vector a are computed by the logarithmic convertor 136 as the logarithms, to the base b, of the elements of the a' vector over GF(m). The operation and structure of a logarithmic convertor 136 is described below.

It is well known that if p is prime then

$$z^{p-1}=1 \pmod p),\ 1\leq z\leq p-1 \tag{15}$$

Consequently arithmetic in the exponent is done modulo p−1, not modulo p. That is

$$z^x=z^{x \bmod p-1} \pmod p \tag{16}$$

for all integers x.

The algorithm for computing logarithms mod p is best understood by first considering the special case $p=2^n+1$. We are given α, p and y, with α a primitive element of GF(p), and must find x such that $y=\alpha^x$ (mod p). We can assume $0\leq x\leq p-2$, since $x=p-1$ is indistinguishable from $x=0$.

When $p=2^n+1$, x is easily determined by finding the binary expansion $\{b_0, \ldots, b_{n-1}\}$ of x. The least significant bit, b_0, of x is determined by raising y to the $(p-1)/2=2^{n-1}$ power and applying the rule

$$y^{(p-1)/2} \pmod p = \begin{cases} +1, & b_0=0 \\ -1, & b_0=1. \end{cases} \tag{17}$$

This fact is established by noting that since α is primitive

$$\alpha^{(p-1)/2}=-1 \pmod p), \tag{18}$$

and therefore

$$y^{(p-1)/2}=(\alpha^x)^{(p-1)/2}=(-1)^x \pmod p). \tag{19}$$

The next bit in the expansion of x is then determined by letting

$$z=y\alpha^{-b_0}=\alpha^{x_1} \pmod p) \tag{20}$$

where

$$x_1=\sum_{i=1}^{n-1} b_i 2^i. \tag{21}$$

Clearly x_1 is a multiple of 4 if and only if $b_1=0$. If $b_1=1$ then x_1 is divisible by 2, but not by 4. Reasoning as before,

$$z^{(p-1)/4} \pmod p = \begin{cases} +1, & b_1=0 \\ -1, & b_1=1. \end{cases} \tag{22}$$

The remaining bits of x are determined in a similar manner. This algorithm is summarized in the flow chart of FIG. 12.

To aid in understanding this flowchart, note that at the start of the i^{th} loop,

$$m=(p-1)/2^{i+1} \tag{23}$$

and

$$z=\alpha^{x_i} \pmod p) \tag{24}$$

where

$$x_i=\sum_{j=i}^{n-1} b_j 2^j. \tag{25}$$

Thus raising z to the m^{th} power gives

$$\begin{aligned} z^m &= \alpha^{(x_i m)} = \alpha^{[(p-1)/2]\,(x_i/2^i)} \\ &= (-1)^{x_i/2^i} = (-1)^{b_i} \pmod p), \end{aligned} \tag{26}$$

so that $z^m=1$ (mod p) if and only if $b_i=0$, and $z^m=-1$ (mod p) if and only if $b_i=1$.

As an example, consider $p=17=2^4+1$. Then $\alpha=3$ is primitive ($\alpha=2$ is not primitive because $2^8=256=1$, mod 17). Given $y=10$ the algorithm computes x as follows (note that $\beta=x^{-1}=6$ since $3\times 6=18=1$, mod 17):

i	Z	β	m	W	b_i
0	10	6	8	16	1
1	9	2	4	16	1
2	1	4	2	1	0
3	1	16	1	1	0
4		1	[illegible]		

It thus finds that $x=2^0+2^1=3$. This is correct because $\alpha^3=3^3=27=10$ (mod 17).

We now generalize this algorithm to arbitrary primes p. Let

$$p-1=p_1^{n_1}p_2^{n_2}\cdots p_k^{n_k},\ p_i<p_{i+1} \tag{27}$$

be the prime factorization of p−1, where the p_i are distinct primes, and the n_i are positive integers. The value of x (mod $p_i^{n_i}$) will be determined for $i=1, \ldots, k$ and the results combined via the Chinese remainder theorem to obtain

$$x \left(\bmod \prod_{i=1}^{k} p_i^{n_i}\right) = x \pmod{p-1} = x \tag{28}$$

since $0\leq x\leq p-2$. The Chinese remainder theorem can be implemented in $O(k \log_2 p)$ operations and $O(k \log_2 p)$ bits of memory. (We count a multiplication mod p as one operation.)

Consider the following expansion of x (mod $p_i^{n_i}$).

$$x \left(\bmod p_i^{n_i}\right) = \sum_{j=0}^{n_i-1} b_j p_i^j \tag{29}$$

where $0\leq b_j\leq p_i-1$.

The least significant coefficient, b_0, is determined by raising y to the $(p-1)/p_i$ power,

$$y^{(p-1)/p_i}=\alpha^{(p-1)x/p_i}=\gamma_i^x=(\gamma_i)^{b_0} \pmod p) \tag{30}$$

where

$$\gamma_i = \alpha^{(p-1)/p_i} \pmod p \quad (31)$$

is a primitive p_i^{th} root of unity. There are therefore only p_i possible values for $y^{(p-1)/p_i}$ (mod p), and the resultant value uniquely determines b_0.

The next digit, b_1, in the base p_i expansion of x (mod $p_i^{n_i}$) is determined by letting

$$z = y \cdot \alpha^{-x_0} = \alpha^{x_1} \pmod p, \quad (32)$$

where

$$x_1 = \sum_{j=1}^{n_i-1} b_j p_i^j \quad (33)$$

Now, raising z to the $(p-1)/p_i^2$ power yields

$$z^{(p-1)/p_i^2} = \alpha^{(p-1)x_1/p_i^2} = \gamma_i^{x_1/p_i} = (\gamma_i)^{b_1} \pmod p. \quad (34)$$

Again, there are only p_i possible values of $z^{(p-1)/p_i^2}$ and this value determines b_1. This process is continued to determine all the coefficients, b_j.

The flow chart of FIG. 13 summarizes the algorithm for computing the coefficients (b_j) of the expansion (29). This algorithm is used k times to compute x (mod $p_i^{n_i}$) for $i=1, 2, \ldots k$, and these results are combined by the Chinese remainder theorem to obtain x. The function $g_i(w)$ in FIG. 13 is defined by

$$\gamma_i^{g_i(w)} = w \pmod p,\ 0 \le g_i(w) \le p_i - 1, \quad (35)$$

where γ_i is defined in (31).

If all prime factors, $\{p_i\}_{i=1}^k$, of p−1 are small then the $g_i(w)$ functions are easily implemented as tables, and computing a logarithm over GF(p) requires $O(\log_2 p)^2$ operations and only minimal memory for the $g_i(w)$ tables. The dominant computational requirement is computing $w = z^n$, which requires $O(\log_2 p)$ operations. This loop is traversed

$$\sum_{i=1}^{k} n_i$$

times, and if all p_i are small,

$$\sum_{i=1}^{k} n_i$$

is approximately $\log_2$ p. Thus when p−1 has only small prime factors it is possible to compute logarithms over GF(p) easily.

As an example, consider $p=19$, $\alpha=2$, $y=10$. Then $p-1=2\cdot 3^2$ and $p_1=2$, $n_1=1$, $p_2=3$ and $n_2=2$. The computation of $x \pmod{p_1^{n_1}} = x \pmod 2$ involves computing $y^{(p-1)/p_1} = \alpha^9 = 512 = 18 \pmod{19}$ so $b_1=1$ and $x \pmod 2 = 2^0 = 1$ (i.e., x is odd). Next the flow chart of FIG. 13 is re-executed for $p_2=3$ $n_2=2$ as follows ($\beta=10$ because $2 \times 10 = 20 = 1$, mod 19; further $\gamma_2 = \alpha^6 = 7$ and $7^0=1$, $7^1=7$, and $7^2=11 \pmod{19}$ so $g_2(1)=0$, $g_2(7)=1$ and $g_2(11)=2$):

Z	B	n	j	W	b_j
10	10	6	0	11	2
12	12	2	1	11	2
18	18	1	2		

so that $x \pmod{p_2^{n_2}} = x \pmod 9 = 2 \cdot 3^0 + 2 \cdot 3^1 = 8$

Knowing that $x \pmod 2 = 1$ and that $x \pmod 9 = 8$ implies that $x \pmod{18} = 17$. (Either the Chinese Remainder Theorem can be used, or it can be realized that $x=8$ or $x=8+9=17$ and only 17 is odd.) As a check we find that $2^{17}=131{,}072=10 \pmod{19}$, so that $y=\alpha^x$ (mod p).

It is seen that the logarithmic convertor requires a mod p inverter for computing $\beta=\alpha^{-1}$ (mod p). As already noted, this can be obtained using the extended form of Euclid's algorithm, which requires the use of the divider of FIG. 8, the multiplier of FIG. 3, and the comparator of FIG. 5. The logarithmic convertor also requires the divider of FIG. 8 (for computing successive values of n), the adder of FIG. 4 (for incrementing j), the modulo p exponentiator of FIG. 10 (for computing W and β^{b_j} and for precomputing the $g_i(W)$ table), the modulo p multiplier of FIG. 3 (for computing successive values of Z), and the comparator of FIG. 5 (for determining when $j=N_i$). The logarithmic convertor's use of the Chinese remainder theorem requires only devices which have already been described (the multiplier of FIG. 3 and a modulo m inverter).

In the first method of generating a trap door knapsack vector, a very difficult knapsack problem involving a vector a was transformed into a very simple and easily solved knapsack problem involving a', by means of the transformation:

$$a_i' = 1/w^* a_i \bmod m \quad (36)$$

A knapsack involving a could be solved because it was transformable into another knapsack involving a' that was solvable. Notice, though, that is does not matter why knapsacks involving a' are solvable. Thus, rather than requiring that a' satisfy (1), it could be required that a' be transformable into another knapsack problem involving a", by the transformation:

$$a_i'' = 1/w'^* a_i' \bmod m' \quad (37)$$

where a" satisfies (1), or is otherwise easy to solve. Having done the transformation twice, there is no problem in doing this a third time; in fact, it is clear that this process may be iterated as often as desired.

With each successive transformation, the structure in the publicly known vector, a, becomes more and more obscure. In essence, we are encrypting the simple knapsack problem by the repeated application of a transformation which preserves the basic structure of the problem. The final result a appears to be a collection of random numbers. The fact that the problem can be easily solved has been totally obscured.

The original, easy to solve knapsack vector can meet any condition, such as (1) which guarantees that it is easy to solve. For example it could be a multiplicative trap door knapsack. In this way it is possible to combine both of the trap door knapsack methods into a single method, which is presumably harder to break.

It is important to consider the rate of growth of a, because this rate determines the data expansion involved in transmitting the n dimensional vector x as the larger quantity S. The rate of growth depends on the method of selecting the numbers, but in a "reasonable" implementation, with $n=100$, each a_i will be at most 7 bits larger than the corresponding a'_i, each a'_i will be at most 7 bits larger than a''_i, etc., etc. Each successive stage of the transformation will increase the size of the problem by only a small, fixed amount. If we repeat the transformation 20 times, this will add at most 140 bits to each a_i. If each a_i is 200 bits long to begin with, then they need only be 340 bits long after 20 stages. The knapsack vector, for $n=100$, will then be at most $100*340=34$ kilobits in size.

Usual digital authenticators protect against third party forgeries, but cannot be used to settle disputes between the transmitter 11 and receiver 12 as to what message, if any, was sent. A true digital signature is also called a receipt because it allows the receiver 12 to prove that a particular message M was sent to it by the transmitter 11. Trap door knapsacks can be used to generate such receipts in the following manner.

If every message M in some large fixed range had an inverse image x, then it could be used to provide receipts. Transmitter 11 creates knapsack vectors b' and b such that b' is a secret key, such as an easily solved knapsack vector, and that b is a public key, such as is obtained via the relation

$$b_i = w^* b_i' \bmod m \tag{38}$$

Vector b is then either placed in a public file or transmitted to receiver 12. When transmitter 11 wants to provide a receipt for message M, transmitter 11 would compute and transmit x such that $b^*x=M$. Transmitter 11 creates x for the desired message M by solving the easily solved knapsack problem.

$$\begin{aligned} M' &= 1/w * M \bmod m & (39)\\ &= 1/w * \Sigma\, x_i * b_i \bmod m & (40)\\ &= 1/w * \Sigma\, x_i * w * b_i' \bmod m & (41)\\ &= \Sigma\, x_i * b_i' \bmod m & (42) \end{aligned}$$

The receiver 12 could easily compute M from x and, by checking a date/time field (or some other redundancy in M), determine that the message M was authentic. Because the receiver 12 could not generate such an x, since it requires b' which only the transmitter 11 possesses, the receiver 12 saves x as proof that transmitter 11 sent message M.

This method of generating receipts can be modified to work when the density of solutions (the fraction of messages M between 0 and Σb_i which have solutions to $b^*x=M$) is less than 1, provided it is not too small. The message M is sent in plaintext form, or encrypted as described above if eavesdropping is a concern, and a sequence of related oneway functions $y_1=F_1(M)$, $y_2=F_2(M)$, . . . are computed. The transmitter 11 then seeks to obtain an inverse image, x, for y_1, y_2, . . . until one is found and appends the corresponding x to M as a receipt. The receiver 12 computes $M'=b^*x$ and checks that $M'=y_i$ where i is within some acceptable range.

The sequence of one-way functions can be as simple as:

$$F_i(M)=F(M)+i \tag{43}$$

or

$$F_i(M)=F(M+i) \tag{44}$$

where F(*) is a one-way function. It is necessary that the range of F(*) have at least 2^{100} values to foil trial and error attempts at forgery.

It is also possible to combine the message and receipt as a single message-receipt datum. If the acceptable range for i is between 0 and 2^I-1 and the message is J

bits long then a single number, J + I bits long, can represent both the message and i. The transmitter 11 checks for a solution to $b^*x=S$ for each of the 2^I values of S which result when, for example, the first J bits of S are set equal to the message and the last I bits of S are unconstrained. The first such solution x is conveyed to the receiver 12 as the message-receipt. Receiver 12 recovers S by computing the dot product of the public key b and the message-receipt combination x, and retaining the first J bits of S thus obtained. The authenticity of the message is validated by the presence of appropriate redundancy in the message, either natural redundancy if the message is expressed in a natural language such as English, or artificial redundancy such as the addition of a date-time field in the message.

Redundancy is used here in the sense of information theory [Claude E. Shannon, "The Mathematical Theory of Communication", *Bell System Technical Journal*, Vol. 27, p. 379 and p. 623, 1948] and complexity theory [Gregory J. Chaitin "On the Length of Programs for Computing Finite Binary Sequences", Journal of the Association for Computing Machinery, Vol. 13, p. 547, 1966] to measure the structure (deviation from complete randomness and unpredictability) in a message. A source of messages possesses no redundancy only if all characters occur with equal probability. If it is possible to guess the characters of the message with a better than random success rate, the source possesses redundancy and the rate at which a hypothetical gambler can make his fortune grow is the quantitative measure of redundancy. [Thomas M. Cover and Roger C. King, "A Convergent Gambling Estimate of the Entropy of English", Technical Report #22, Statistics Department, Stanford University, Nov. 1, 1976]. Humans can easily validate the message by performing a redundancy check (e.g., determine if the message is grammatically correct English). By simulating the gambling situation, it is possible for a machine to validate whether or not a message possesses the redundancy appropriate to its claimed source.

There are many methods for implementing this form of the invention. Part of the deciphering key D could be public knowledge rather than secret, provided the part of D which is withheld prevents the eavesdropper 13 from recovering the plaintext message X.

Variations on the above described embodiment are possible. For example, in some applications, it will prove valuable to have the ith receiver of the system generate a trap door knapsack vector $a^{(i)}$ as above, and place the vector or an abbreviated representation of the vector in a public file or directory. Then, a transmitter who wishes to establish a secure channel will use $a^{(i)}$ as the enciphering key for transmitting to the ith receiver. The advantage is that the ith receiver, once having proved his identity to the system through the use of his driver's license, fingerprint, etc., can prove his identity to the transmitter by his ability to decipher data encrypted with enciphering key $a^{(i)}$. Thus, although the best mode contemplated for carrying out the present invention has been herein shown and described, it will be apparent that modification and variation may be made without departing from what is regarded to be the subject matter of this invention.

What is claimed is:

1. In a method of communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

providing random numbers at the receiver;

generating from said random numbers a public enciphering key at the receiver;

generating from said random numbers a secret deciphering key at the receiver such that the secret deciphering key is directly related to and computationally infeasible to generate from the public enciphering key;

communicating the public enciphering key from the receiver to the transmitter;

processing the message and the public enciphering key at the transmitter and generating an enciphered message by an enciphering transformation, such that the enciphering transformation is easy to effect but computationally infeasible to invert without the secret deciphering key;

transmitting the enciphered message from the transmitter to the receiver; and

processing the enciphered message and the secret deciphering key at the receiver to transform the enciphered message with the secret deciphering key to generate the message.

2. In a method of communicating securely over an insecure communication channel as in claim 1, further comprising:

authenticating the receiver's identity to the transmitter by the receiver's ability to decipher the enciphered message.

3. In a method of communicating securely over an insecure communication channel as in claim 2 wherein the step of:

authenticating the receiver's identity includes the receiver transmitting a representation of the message to the transmitter.

4. In a method of providing a digital signature for a communicated message comprising the steps of

providing random numbers at the transmitter;

generating from said random numbers a public key at the transmitter;

generating from said random numbers a secret key at the transmitter such that the secret key is directly related to and computationally infeasible to generate from the public key;

processing the message to be transmitted and the secret key at the transmitter to generate a digital signature at said transmitter by transforming a representation of the message with the secret key, such that the digital signature is computationally infeasible to generate from the public key;

communicating the public key to the receiver;

transmitting the message and the digital signature from the transmitter to the receiver;

receiving the message and the digital signature at the receiver and transforming said digital signature with the public key to generate a representation of the message; and

validating the digital signature by comparing the similarity of the message to the representation of the message generated from the digital signature.

5. A method of providing a message digital signature receipt for a communicated message comprising the steps of:

providing random numbers at the transmitter;

generating from said random numbers a public key at the transmitter;

generating from said random numbers a secret key at the transmitter such that the secret key is directly related to and computationally infeasible to generate from the public key;

processing the message and the secret key at the transmitter and generating a message-digital signature at said transmitter by transforming a representation of the message with the secret key, such that the message-digital signature is computationally infeasible to generate from the public key;

communicating the public key to the receiver;

transmitting the message-digital signature from the transmitter to the receiver;

processing the message-digital signature and the public key at the receiver and transforming the message-digital signature with the public key; and

validating the transformed message-digital signature by checking for redundancy.

6. In an apparatus for communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

means for generating random information at the receiver;

means for generating from said random information a public enciphering key at the receiver, means for generating from said random information a secret deciphering key such that the secret deciphering key is directly related to and computationally infeasible to generate from the public enciphering key;

means for communicating the public enciphering key from the receiver to the transmitter;

means for enciphering a message at the transmitter having an input connected to receive said public enciphering key, having another input connected to receive said message, and serving to transform said message with said public enciphering key, such that the enciphering transformation is computationally infeasible to invert without the secret deciphering key;

means for transmitting the enciphered message from the transmitter to the receiver; and

means for deciphering said enciphered message at the receiver having an input connected to receive said enciphered message, having another input connected to receive said secret deciphering key and serving to generate said message by inverting said transformation with said secret deciphering key.

7. In a method of communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

generating a secret deciphering key at the receiver by generating an n dimensional vector a', the elements of vector a', being defined by

$$a_i' > \sum_{j=1}^{i-1} a_j' \text{ for } i = 1, 2, \ldots n$$

where n is an integer;

generating a public enciphering key at the receiver by generating an n dimensional vector a, the elements of vector a being defined by

$$a_i = (w^* a_i' \bmod m) + km \text{ for } i = 1, 2, \ldots n$$

where m and w are large integers, w is invertible modulo m, and k is an integer;

P018947

transmitting the public enciphering key from the receiver to the transmitter;

receiving the message and the public enciphering key at the transmitter and generating an enciphered message by computing the dot product of the message, represented as a vector x with each element being 0 or 1, and the public enciphering key, represented as the vector a, to represent the enciphered message S being defined by

$$S = a^{*}x$$

transmitting the enciphered message from the transmitter to the receiver; and

receiving the enciphered message and the secret deciphering key at the receiver and transforming the enciphered message with the secret deciphering key to generate the message by computing

$$S' = 1/w^{*}S \bmod m$$

and letting $x_i = 1$ if and only if

$$[S' - \sum_{j=i+1}^{n} x_j * a_j'] \geq a_i'$$

and letting $x_i = 0$ if

$$[S' - \sum_{j=i+1}^{n} x_j * a_j'] < a_i'$$

for $i = n, n-1, \ldots 1$.

8. In a method of communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

generating a secret deciphering key at the receiver by generating an n dimensional vector a', the elements of vector a' being defined by

$$a_i' > l \sum_{j=1}^{i-1} a_j'$$

for i=1, 2, . . . n

where l and n are integers;

generating a public enciphering key at the receiver by generating an n dimensional vector a, the elements of vector a being defined by

$$a_i = (W^{*}a_i' \bmod m) + km \text{ for } i = 1, 2, \ldots n$$

where m and w are large integers, w is invertible modulo m and k is an integer;

transmitting the public enciphering key from the receiver to the transmitter;

receiving the message and the public enciphering key at the transmitter and generating an enciphered message by computing the dot product of the message, represented as a vector x with each element being an integer between 0 and l, and the public enciphering key, represented as the vector a, to represent the enciphered message S being defined by

$$S = a^{*}x;$$

transmitting the enciphered message from the transmitter to the receiver; and

receiving the enciphered message and the secret deciphering key at the receiver and transforming the enciphered message with the secret deciphering key to generate the message by computing

$$S' = 1/w^{*}S \bmod m$$

and letting x_i be the integer part of

$$[S' - \sum_{j=i+1}^{n} x_j * a_j']/a_i'$$

for $i = n, n-1, \ldots 1$.

9. In a method of communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

generating a secret deciphering key at the receiver by generating an n dimensional vector a', the elements of vector a' being relatively prime and n being an integer;

generating a public enciphering key at the receiver by generating an n dimensional vector a, the elements of vector a being defined by

$$a_i = \log_b a_i' \bmod m \text{ for } i = 1, 2 \ldots n$$

where b and m are large integers and m is a prime number such that

$$m > \prod_{i=1}^{n} a_i';$$

transmitting the public enciphering key from the receiver to the transmitter;

receiving the message and the public enciphering key at the transmitter and generating an enciphered message by computing the dot product of the message, represented as a vector x, and the public enciphering key, represented as the vector a, to represent the enciphered message S being defined by

$$S = a^{*}x;$$

transmitting the enciphered message from the transmitter to the receiver; and

receiving the enciphered message and the secret deciphering key at the receiver and transforming the enciphered message with the secret deciphering key to generate the message by computing

$$S' = b^{S} \bmod m$$

and letting $x_i = 1$ if and only if the quotient of S'/a_i' is an integer and letting $x_i = 0$ if the quotient of S'/a_i' is not an integer.

10. In an apparatus for communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

means for generating a secret deciphering key at the receiver by generating an n dimension vector a', the elements of vector a' being defined by

$$a'_i > \sum_{j=1}^{i-1} a'_j \text{ for } i = 1, 2, \ldots n$$

where n is an integer:

means for generating a public enciphering key at the receiver by generating an n dimensional vector a, the elements of vector a being defined by

$$a_i = (w^* a'_i \bmod m) + km \text{ for } i = 1, 2, \ldots, n$$

where m and w are large integers, w is invertible modulo m, and k is an integer;

means for transmitting the public enciphering key from the receiver to the transmitter;

means, for enciphering a message at the transmitter, having an input connected to receive the public enciphering key, having another input connected to receive the message, and having an output that generates an enciphered message that is a transformation of the message with the public enciphering key by computing the dot product of the message, represented as a vector x with each element being 0 or 1, and the public enciphering key, represented as the vector a, to represent the enciphered message S being defined by

$$S = a^* x$$

means for transmitting the enciphered message from the transmitter to the receiver; and

means for deciphering the enciphered message at the receiver, having an input connected to receive the enciphered message, having aother input connected to receive the secret deciphering key, and having an output for generating the message by inverting the transformation with the secret deciphering key by computing

$$S' = 1/w^* S \bmod m$$

and letting $x_i = 1$ if and only if

$$[S' - \sum_{j=i+1}^{n} x_j{}^* a'_j] \geq a'_i$$

and letting $x_i = 0$ if

$$[S' - \sum_{j=i+1}^{n} x_j{}^* a'_j] < a'_i$$

for $i = n, n-1, \ldots, 1$.

11. In an apparatus for communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

means for generating a secret deciphering key at the receiver by generating an n dimensional vector a', the elements of vector a' being defined by

$$a'_i > l \sum_{j=1}^{i-1} a'_j \text{ for } i = 1, 2, \ldots n$$

where l and n are integers;

means for generating a public enciphering key at the receiver by generating an n dimensional vector a, the elements of vector a being defined by

$$a_i = (w^* a'_i \bmod m) + km \text{ for } i = 1, 2 \ldots n$$

where m and w are large integers, w is invertible modulo m, and k is an integer;

means for transmitting the public enciphering key from the receiver to the transmitter;

means, for enciphering a message at the transmitter, having an input connected to receive the public enciphering key, having another input connected to receive the message, and having an output that generates an enciphered message that is a transformation of the message with the public enciphering key by computing the dot product of the message, represented as a vector x with each element being an integer between 0 and l, and the public enciphering key, represented as the vector a, to represent the enciphered message S being defined by

$$S = a^* x$$

means for transmitting the enciphered message from the transmitter to the receiver; and

means for deciphering the enciphered message at the receiver, having an input connected to receive the enciphered message, having another input connected to receive the secret deciphering key, and having an output for generating the message by inverting the transformation with the secret deciphering key by computing

$$S' = 1/w^* S \bmod m$$

and letting x_i be the integer part of

$$[S' - \sum_{j=i+1}^{n} x_j{}^* a'_j]/a'_i$$

for $i = n, n-1, \ldots, 1$.

12. In an apparatus for communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

means for generating a secret deciphering key at the receiver by generating an n dimensional vector a', the elements of vector a' being relatively prime and n being an integer;

means for generating a public enciphering key at the receiver by generating an n dimensional vector a, the elements of vector a being defined by

$$a_i = \log_b a'_i \bmod m \text{ for } i = 1, 2, \ldots n$$

where b and m are large integers and m is a prime number such that

$$m > \prod_{i=1}^{n} a'_i;$$

means for transmitting the public enciphering key from the receiver to the transmitter;

means, for enciphering a message at the transmitter, having an input connected to receive the public enciphering key, having another input connected to receive the message, and having an output that generates an enciphered message that is a transformation of the message with the public enciphering key by computing the dot product of the message, represented as a vector x with each element being 0 or 1, and the public enciphering key, represented as the vector a, to represent the enciphered message S being defined by

$$S = a * x;$$

means for transmitting the enciphered message from the transmitter to the receiver; and

means for deciphering the enciphered message at the receiver, having an input connected to receive the enciphered message, having another input connected to receive the secret deciphering key, and having an output for generating the message by inverting the transformation with the secret deciphering key by computing

$$S' = b^S \bmod m$$

and letting $x_i = 1$ if and only if the quotient of S'/a_i is an integer and letting $x_i = 0$ of the quotient of S'/a_i is not an integer.

13. In an apparatus for enciphering a message that is to be transmitted over an insecure communication channel having an input connected to receive a message to be maintained secret, having another input connected to receive a public enciphering key, and having an output for generating the enciphered message, characterized by:

means for receiving the message and converting the message to a vector representation of the message;

means for receiving the public enciphering key and converting the public enciphering key to a vector representation of the public enciphering key; and

means for generating the enciphered message by computing the dot product of the vector representation of the message and the vector representation of the public enciphering key, having an input connected to receive the vector representation of the message, having another input connected to receive the vector representation of the public enciphering key, and having an output for generating the enciphered message.

14. In a method of communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver the improvement characterized by:

generating a secret deciphering key at the receiver;

generating a public enciphering key at the receiver, such that the secret deciphering key is computationally infeasible to generate from the public enciphering key;

transmitting the public enciphering key from the receiver to the transmitter;

processing the message and the public enciphering key at the transmitter by computing the dot product of the message, represented as a vector, and the public enciphering key, represented as a vector, to represent the enciphered message, such that the enciphering transformation is easy to effect but computationally infeasible to invert without the secret deciphering key;

transmitting the enciphered message from the transmitter to the receiver;

and processing the enciphered message and the secret deciphering key at the receiver and inverting said transformation by transforming the enciphered message with the secret deciphering key to generate the message.

15. In an apparatus for communicating securely over an insecure communication channel of the type which communicates a message from a transmitter to a receiver, the improvement characterized by:

means for generating a secret deciphering key at the receiver;

means for generating a public enciphering key at the receiver, such that the secret deciphering key is computationally infeasible to generate from the public enciphering key;

means for transmitting the public enciphering key from the receiver to the transmitter;

means for enciphering a message at the transmitter having an input connected to receive said public enciphering key and having another input connected to receive said message and serving to transform said message by computing the dot product of said message, represented as a vector, and said public enciphering key, represented as a vector, to represent said enciphered message, such that the enciphering transformation is computationally infeasible to invert without the secret deciphering key;

means for transmitting the enciphered message from the transmitter to the receiver;

and means for deciphering said enciphered message at the receiver, said means having an input connected to receive said enciphered message and having another input connected to receive said secret deciphering key and serving to generate said message by inverting the transformation with said secret deciphering key.

16. An apparatus for deciphering an enciphered message that is received over an insecure communication channel including

means for receiving the enciphered message that is enciphered by an enciphering transformation in which a message to be maintained secret is transformed with a public enciphering key, and means for receiving a secret deciphering key to generate the message by inverting the enciphering transformation;

means for generating the message having an input connected to receive the inverse of the enciphered message and an output for generating the message;

said secret deciphering key being computationally infeasible to generate from the public enciphering key, and said enciphering transformation being computationally infeasible to invert without the secret deciphering key in which

said means for inverting the enciphering transformation includes means for computing

$$S' = 1/w * S \bmod m; \text{ and}$$

said means for generating the message includes means for setting x_i equal to the integer part of

$$\left[S' - \sum_{j=i+1}^{n} x_j * a_j' \right] / a_i' \text{ for } i = n, n-1, \ldots 1$$

where m and w are large integers and w is invertible modulo m, where S' is the inverse of the enciphered message S being defined by the enciphering transformation

P018950

$$S = a * x$$

where the message is represented as an n dimensional vector x with each element x_i being an integer between 0 and l, where l is an integer, and where the public enciphering key is represented as an n dimensional vector a, the elements of a being defined by

$$a_i = (w^* a_i' \bmod m) + km \text{ for } i = 1, 2, \ldots, n$$

where k and n are integers and the secret deciphering key is m, w and a', where a' is an n dimensional vector, the elements of a' being defined by

$$a_i' > l \sum_{j=1}^{i-1} a_j' \text{ for } i = 1, 2, \ldots, n$$

17. An apparatus for deciphering an enciphered message that is received over an insecure communication channel including

means for receiving the enciphered message that is enciphered by an enciphering transformation in which a message to be maintained secret is transformed with a public enciphering key, and means for receiving a secret deciphering key to generate the messge by inverting the enciphering transformation;

means for generating the message having an input connected to receive the inverse of the enciphered message and an output for generating the message;

said secret deciphering key being computationally infeasible to generate from the public enciphering key, and said enciphering transformation being computationally infeasible to invert without the secret deciphering key in which

said means for inverting the enciphering transformation includes means for computing

$$S' = b^S \bmod m, \text{ and}$$

said means for generating the message includes means for setting $x_i = 1$ if and only if the quotient of S'/a_i' is an integer and setting $x_i = 0$ of the quotient of S'/a_i' is not an integer, where b and m are large integers and m is a prime number such that

$$m > \prod_{i=1}^{n} a_i'$$

where n is an integer and the secret deciphering key is b,m, and a', where a' is an n dimensional vector with each element a_i' being relatively prime, and where S' is the inverse of the enciphered message S being defined by the enciphering transformation

$$S = a * x$$

where the message is represented as an n dimensional vector x with each element x_i being 0 or 1, and the public enciphering key is represented as the n dimensional vector a, the elements of a being defined by

$$a_i = \log_b a_i' \bmod m \text{ for } i = 1, 2, \ldots, n$$

* * * * *

P018951

C

Multiuser cryptographic techniques*

by WHITFIELD DIFFIE and MARTIN E. HELLMAN
Stanford University
Stanford, California

ABSTRACT

This paper deals with new problems which arise in the application of cryptography to computer communication systems with large numbers of users. Foremost among these is the key distribution problem. We suggest two techniques for dealing with this problem. The first employs current technology and requires subversion of several separate key distribution nodes to compromise the system's security. Its disadvantage is a high overhead for single message connections. The second technique is still in the conceptual phase, but promises to eliminate completely the need for a secure key distribution channel, by making the sender's keying information public. It is also shown how such a public key cryptosystem would allow the development of an authentication system which generates an unforgeable, message dependent digital signature.

INTRODUCTION

In a computer network with a large number of users, cryptography is often essential for protecting stored or transmitted data. While this application closely resembles the age old use of cryptography to protect military and diplomatic communications, there are several important differences which require new protocols and new types of cryptosystems. This paper addresses the multiuser aspect of computer networks and presents ways to preserve privacy of communication despite the large number of user connections which are possible.

In a system with n users there are n^2-n pairs who may wish to hold private conversations. The straightforward way to achieve this is to give each pair of users a key in common which they share with no one else. Each user will then have n-1 keys, one for communicating with each other user. Unfortunately, the cost of distributing these keys is prohibitive. A new user must send keys to all other users. Unfortunately, the network cannot be used for this purpose, and an external secure channel is required. This procedure is comparable to requiring each new telephone subscriber to send a registered letter to everyone else in the phonebook.

Military communications suffer less from this problem for several reasons. Among these are the limitations imposed by the chain of command and the fact that stations change allegiance infrequently. In a computer network designed for business communication, on the other hand, users will regard each other as friends on one matter and as opponents on another. Firms A and B may cooperate on one venture in competition with C, while simultaneously, A and C compete with B on a different endeavor. A must therefore use different keys for communicating with B and C.

One approach to this problem is to assume that the users trust the network. Each user remembers only one key which is used to communicate with a local node. From there the message is relayed from node to node, each of which decrypts it, then reencrypts it in a different key for the next leg of its journey. This process is known as link encryption.[1] When the message reaches the network node closest to its destination, it is sent on to the addressee encrypted in a key shared only by the addressee and that node.

Although this technique requires each user to remember only one key, it has the disadvantage that a message is compromised if any one of the nodes in its path is subverted. In this paper we examine two other ways of allowing secure communication between any pair of users without assuming the integrity of all nodes in the network and without requiring the users to distribute or store large numbers of keys.

The first technique requires no new technology, but imposes a complex initial connection protocol. This is the subject of the second section of this paper. We call the second technique public key cryptography, since most of the secrecy traditionally required for the keys has been removed. This is discussed in section three and represents a radical departure from past cryptographic practices. While it requires further work before it becomes implementable its simplicity of operation makes it extremely attractive. If a suc-

* This work was supported by the National Science Foundation under NSF Grant ENG 10173.

cessful implementation can be developed it should find wide use in both military and civilian applications.

The fourth section shows how public key cryptography can be used to provide a time and message dependent digital signature which cannot be forged even when past signatures have been seen. This is an example of the general problem of authentication discussed in greater detail in Reference 2, which provides a more general perspective in which public key cryptography can be viewed.

A PROTECTIVE PROTOCOL

As indicated earlier, a message protected by link encryption will be compromised if any node in the path it follows from the sender to the receiver is subverted. In this section we describe a protocol which guarantees to protect the message unless a large number of nodes are compromised. While many variations are possible, the basic technique is as follows.

A small number m of the network's nodes will function as "key distribution nodes." Each user has m keys, one for communicating with each of these m nodes. These keys vary from user to user, so while each user must remember only m keys, each of the key distribution nodes remembers n, one for each user of the net. When users A and B wish to establish a secure connection they contact the m key distribution nodes and receive one randomly chosen key from each. These keys are sent in encrypted form using the keys which the users share with the respective nodes. Upon receiving these keys, the conversants each compute the exclusive or of the m keys received to obtain a single key which is then used to secure a private conversation. None of the nodes involved can violate this privacy individually. Only if all m nodes are compromised will the security of this connection fail.

It might be objected that any key distribution node acting alone can prevent all communication by mischievously sending out different keys to each of the parties, thus bringing network operations to a halt. The users, however, can easily protect themselves against this threat. If communication using the composite key fails, its use as a key is abandoned, and the components are exchanged one by one, in clear, for comparison. If any key fails to agree, the node which issued it is blacklisted. Finally, on conclusion of this process, the users repeat the request for keys to the nodes which passed the previous test.

Alternatively, the component keys can be compared by the use of one way functions[2,3,5] without ever being transmitted in clear. Loosely speaking, a function f is called a one-way function if it is easy to compute in the forward direction, but given any output, it is computationally infeasible to find an input which produces it. In referring to a task as computationally infeasible, we have in mind that it cannot be done in fewer than a finite but astronomical number of operations, say 2↑100. For practical purposes, this is equivalent to being incomputable. As shown in Reference 2, a one way function can easily be obtained from a secure cryptosystem.

If communication fails using the composite key, the users send the images of the individual keys under a public one-way function. If the image received does not agree with that computed by applying f to the key, the node which issued it is guilty of compromise. Since the valid keys have not been publicly revealed in this process, there is no need to request new ones from the uncompromised nodes. Instead the invalid ones are omitted and the remainder xored.

To sum up, this technique requires each user to remember m keys and each key distribution node to remember n keys. Unless all m key distribution nodes are subverted, any two users can establish a private link through use of a set-up protocol usually requiring 2m exchanges (more are required if a key distribution node has been subverted). The next section describes a concept which eliminates much of this overhead and does not require the user to trust any node. This new concept, if successfully implemented, will make the technique described above obsolete.

PUBLIC KEY CRYPTOGRAPHY

In this section we propose that it is possible to eliminate most of the secrecy surrounding the key used in a communication, and yet to preserve the secrecy of the communication. This is accomplished by giving each user a pair of keys E and D. E is an enciphering key and is public information. D is the corresponding deciphering key, and while this must be kept secret, it need never be communicated, eliminating the need for a secure key distribution channel. Although D is determined by E, it is infeasible to compute D from E.

For reasons of security, generation of this E-D pair is best done at the user's terminal which is assumed to have some computational power. The user then keeps the deciphering key D secret but makes the enciphering key E public by placing it in a central file along with his name and address. Anyone can then encrypt a message and send it to the user, but only the intended receiver can decipher it. Public key cryptosystems can therefore be regarded as multiple access ciphers.

By regularly checking the file of enciphering keys the user can guard against any attempt to alter it surreptitiously. Any such mischief is reported and settled by other authentication means, such as personal appearance.

The crucial feature of a public key system is that it is relatively easy to generate an E-D pair, preferably automatically through a publicly available transformation from a random bit string to E-D, and yet it is computationally infeasible to compute D from E.

At present we have neither a proof that public key systems exist, nor a demonstration system. We hope to have a demonstration E-D pair in the near future, and expect that if the demonstration pair successfully resists attack then we will be able to design an algorithm for automatically generating E-D pairs of a similar kind. In the meantime, the following reasoning is given to help dispel any doubts the reader may have.

A suggestive example is to let the cryptogram, represented as a binary n-vector c equal E m; where m is the message also represented as a binary n-vector, and E is an arbitrary n-by-n invertible matrix. Letting D=E↑-1 we have m=D c. Thus both enciphering and deciphering are easily accomplished with about n↑2 operations. Calculation of D from E, however, involves a matrix inversion which is a harder problem. And it is at least conceptually simpler to obtain an arbitrary pair of inverse matrices than it is to invert a given matrix. Start with the identity matrix I and do elementary row and column operations to obtain an arbitrary invertible matrix E. Then starting with I do the inverses of these same elementary operations in reverse order, to obtain D=E↑-1. The sequence of elementary operations could easily be generated from a random bit string.

Unfortunately, matrix inversion takes only about n↑3 operations even without knowledge of the sequence of elementary operations. The ratio of "cryptanalytic" time (i.e., computing D from E) to enciphering or deciphering time is thus at most n. To obtain ratios of 10↑6 or greater would thus require enormous block sizes. Also, it does not appear that knowledge of the elementary operations used to obtain E from I greatly reduces the time for computing D. And, since there is no round-off error in binary arithmetic numerical stability is of no consequence in the matrix inversion. In spite of its lack of practical utility, this matrix oriented example is still useful for clarifying the relationships necessary in a public key system.

A more practical direction uses the observation that we are really seeking a pair of easily computed inverse algorithms E and D, but that D must be hard to infer from E. This is not as impossible as it may sound. Anyone who has tried to determine what operation is accomplished by someone else's machine language program knows that E itself (i.e., what E does) can be hard to infer from E (i.e., a listing of E). If the program were to be made purposefully confusing through addition of unneeded variables, statements and outputs, then determining an inverse algorithm could be made very difficult indeed. Of course, E must be complicated enough to prevent its identification from input-output pairs.

Another idea appears more promising. Suppose we start with a schematic of a 100 bit input, 100 bit output circuit which merely is a set of 100 wires implementing the identity mapping. Select 4 points in the circuit at random, break these wires, and insert AND, OR and NOT gates which implement a randomly chosen 4 bit to 4 bit invertible mapping (a 4 bit S box in Feistel's notation).[4] Then repeat this insertion operation approximately 100 times to obtain an enciphering circuit E. Knowing the sequence of operations which led to the final E circuit allows one to easily design an inverse circuit D. If however the gates are now randomly moved around on the schematic of E to hide their associations into S boxes, an opponent would have great difficulty in reconstructing the simple description of E in terms of S boxes, and therefore would have great difficulty in constructing a simple version of D. His task could be further complicated by using reduction techniques (e.g. Carnaugh maps) or expansion techniques (e.g. ~(AB)=~A or ~B, or expressing a logical variable in terms of previous variables), and by adding additional, unneeded S boxes and outputs.

For ease of exposition, we have described the implementation of a specific key in hardware. In practice, a special purpose simulator is obviously of most interest. The hardware description is also valuable in exemplifying a generally useful idea. To build a good public key cryptosystem one needs easily inverted elementary building blocks and a general framework for describing the concatenation of these elementary blocks. Here the elementary building blocks are S boxes and the general framework is the schematic diagram. The general framework must also hide the sequence of elementary building blocks so that no one other than the designer can easily implement the sequence of inverse elementary operations. Examination will show that the matrix example had a similar structure, except there the general class of transformations obtainable was too small.

While the above arguments only provide plausibility as opposed to proof, we hope they will stimulate additional work on this promising area of research.

PUBLIC KEY AUTHENTICATION

The purpose of a cryptographic system is to prevent the unauthorized extraction of information from a public (i.e., insecure) channel. The dual problem of authentication is to prevent unauthorized injection of messages into a public channel.

In conventional paper oriented business transactions, signatures provide a generally accepted level of authentication. As electronic communication replaces mail service the need for a digital signature will be strongly felt.

Various types of authentication are now possible, but the development of public key cryptosystems would allow an entirely new dimension.

Currently, most message authentication consists of appending an authenticator pattern, known only to the transmitter and intended receiver, to each message and encrypting the combination. This protects against an eavesdropper being able to forge new, properly authenticated messages unless he has also stolen the key being used. There is no protection against such an eavesdropping thief or against the threat of dispute. That is, the transmitter may transmit a properly authenticated message, later deny this action, and falsely blame the receiver for taking unauthorized action. Or, conversely, the receiver may take unauthorized action, forge a message to itself and then falsely blame the transmitter for these actions. For example, a dishonest stockbroker may try to cover up unauthorized buying and selling for personal gain by forging orders from clients. Or a client may disclaim an order, actually authorized by him, but which is later seen to cause a loss. We will introduce concepts which would allow the receiver to easily verify the authenticity of a message, but which prevent him from generating apparently authenticated messages, thereby protecting against both the threat of eavesdropping thieves and the threat of dispute. Note that these techniques thus provide stronger protection than signatures, voiceprints, etc. which can be forged once seen and are not message dependent.

To obtain an unforgeable digital signature from a public key cryptosystem, the protocol would be as follows: Assume user A wishes to send a message M to user B. The transformed message C=EbDa(M) is sent, where Eb represents the transformation effected by use of B's public enciphering key and Da represents the transformation effected by use of A's secret deciphering key. Upon receipt of C, user B operates first with his secret operation Db and then with the public operation Ea thereby obtaining EaDb(C) = EaDbEbDa (M) = M. No one else can extract M because of the need to know Db. By saving the intermediate result Db(C) = Da(M) user B (and only user B) can prove that he received the specific message M from user A. There must be some structure to the message (e.g., it could include a date and time field) to prevent injection of random bit patterns for C, with the hope that the resultant decoded "message", EaDb(C), might cause random mischief such as deletion of files.

Note that since there is no need for a secure channel for distribution of authentication information, we have a public key authentication system. This system protects against "eavesdropping thieves" and against a dispute as to whether or not an action taken by the receiver was authorized by the transmitter. Similarly, a public key cryptosystem can be used to protect against the other type of dispute in which the transmitter A claims to have issued an order which was not carried out by the receiver B. The transmitter requests that the receiver B send EaDb(M) as a receipt for the message M. By operating on this receipt with his secret operation Da, the transmitter obtains Db (M), which could only have been generated by the receiver B. Only user A can generate this receipt since it requires knowledge of Da.

While the above discussion centered on message authentication it also applies to user authentication. The implicit message becomes "I am user X and the time is T." Inclusion of the time field prevents an eavesdropper from using old authentication signals to pose as someone else. For reasons noted in Reference 2, such a system deserves to be called a one-way IFF system.

We thus see that public key cryptosystems developed for ensuring the privacy of communications, could also be used to ensure their authenticity. They could therefore be used to fill the need for a digital equivalent of a signature. This need is currently a major barrier to the use of electronic mail for business communications, and provides additional motivation for study of public key cryptosystems.

ACKNOWLEDGMENT

The authors wish to thank Leslie Lamport of Massachusetts Computer Associates for several valuable discussions. In particular, the technique described in Section 2 was discovered during one of these conversations.

REFERENCES

1. Baran, Paul. *On Distributed Communications: IX. Security, Secrecy, and Tamper-Free Considerations*, Santa Monica, CA: The Rand Corporation August 1964, (RM-3765-PR).
2. Diffie, Whitfield and Martin E. Hellman, forthcoming paper to be submitted to the *IEEE Transactions on Information Theory*.
3. Evans, Arthur, Jr., William Kantrowitz and Edwin Weiss, "A User Authentication System not Requiring Secrecy in the Computer," *Communications of the ACM*, Vol. 17 No. 8, August 1974, pp. 437-442.
4. Feistel, Horst, "Cryptography and Computer Privacy," *Scientific American*, Vol. 228, No. 5, May 1973, pp. 15-23.
5. Purdy, George B., "A High Security Log-in Procedure," *Communications of the ACM*, Vol. 17, No. 8, August 1974, pp. 442-445.

D

New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

***Abstract*—Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.**

I. Introduction

We stand today on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

The development of computer controlled communication networks promises effortless and inexpensive contact between people or computers on opposite sides of the world, replacing most mail and many excursions with telecommunications. For many applications these contacts must be made secure against both eavesdropping and the injection of illegitimate messages. At present, however, the solution of security problems lags well behind other areas of communications technology. Contemporary cryptography is unable to meet the requirements, in that its use would impose such severe inconveniences on the system users, as to eliminate many of the benefits of teleprocessing.

Manuscript received June 3, 1976. This work was partially supported by the National Science Foundation under NSF Grant ENG 10173. Portions of this work were presented at the IEEE Information Theory Workshop, Lenox, MA, June 23–25, 1975 and the IEEE International Symposium on Information Theory in Ronneby, Sweden, June 21–24, 1976.

W. Diffie is with the Department of Electrical Engineering, Stanford University, Stanford, CA, and the Stanford Artificial Intelligence Laboratory, Stanford, CA 94305.

M. E. Hellman is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305.

The best known cryptographic problem is that of privacy: preventing the unauthorized extraction of information from communications over an insecure channel. In order to use cryptography to insure privacy, however, it is currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such as private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channels without compromising the security of the system. In a *public key cryptosystem* enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enciphered in such a way that only the intended receiver is able to decipher it. As such, a public key cryptosystem is a multiple access cipher. A private conversation can therefore be held between any two individuals regardless of whether they have ever communicated before. Each one sends messages to the other enciphered in the receiver's public enciphering key and deciphers the messages he receives using his own secret deciphering key.

We propose some techniques for developing public key cryptosystems, but the problem is still largely open.

Public key distribution systems offer a different approach to eliminating the need for a secure key distribution channel. In such a system, two users who wish to exchange a key communicate back and forth until they arrive at a key in common. A third party eavesdropping on this exchange must find it computationally infeasible to compute the key from the information overheard. A possible solution to the public key distribution problem is given in Section III, and Merkle [1] has a partial solution of a different form.

A second problem, amenable to cryptographic solution, which stands in the way of replacing contemporary busi-

of pri-
orma-
nel. In
er, it is
share
send-
uch as
sation
a com-
alistic
d long
neans.
oution
isiness

aitting
annels
a. In a
ing are
puting
quiring
hus be
hering
ace his
ny user
r enci-
is able
em is a
there-
less of
ch one
eiver's
s he re-

olic key
n.
ent ap-
ibution
change
ve at a
his ex-
ompute
le solu-
iven in
of a dif-

olution,
ry busi-

ness communications by teleprocessing systems is authentication. In current business, the validity of contracts is guaranteed by signatures. A signed contract serves as legal evidence of an agreement which the holder can present in court if necessary. The use of signatures, however, requires the transmission and storage of written contracts. In order to have a purely digital replacement for this paper instrument, each user must be able to produce a message whose authenticity can be checked by anyone, but which could not have been produced by anyone else, even the recipient. Since only one person can originate messages but many people can receive messages, this can be viewed as a broadcast cipher. Current electronic authentication techniques cannot meet this need.

Section IV discusses the problem of providing a true, digital, message dependent signature. For reasons brought out there, we refer to this as the one-way authentication problem. Some partial solutions are given, and it is shown how any public key cryptosystem can be transformed into a one-way authentication system.

Section V will consider the interrelation of various cryptographic problems and introduce the even more difficult problem of trap doors.

At the same time that communications and computation have given rise to new cryptographic problems, their offspring, information theory, and the theory of computation have begun to supply tools for the solution of important problems in classical cryptography.

The search for unbreakable codes is one of the oldest themes of cryptographic research, but until this century all proposed systems have ultimately been broken. In the nineteen twenties, however, the "one time pad" was invented, and shown to be unbreakable [2, pp. 398–400]. The theoretical basis underlying this and related systems was put on a firm foundation a quarter century later by information theory [3]. One time pads require extremely long keys and are therefore prohibitively expensive in most applications.

In contrast, the security of most cryptographic systems resides in the computational difficulty to the cryptanalyst of discovering the plaintext without knowledge of the key. This problem falls within the domains of computational complexity and analysis of algorithms, two recent disciplines which study the difficulty of solving computational problems. Using the results of these theories, it may be possible to extend proofs of security to more useful classes of systems in the foreseeable future. Section VI explores this possibility.

Before proceeding to newer developments, we introduce terminology and define threat environments in the next section.

II. Conventional Cryptography

Cryptography is the study of "mathematical" systems for solving two kinds of security problems: privacy and authentication. A privacy system prevents the extraction of information by unauthorized parties from messages

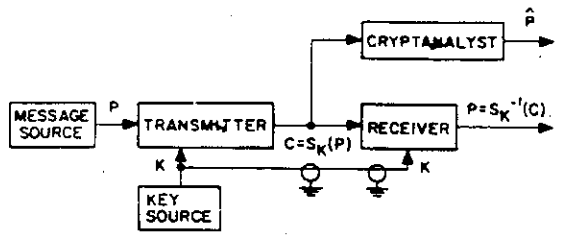


Fig. 1. Flow of information in conventional cryptographic system.

transmitted over a public channel, thus assuring the sender of a message that it is being read only by the intended recipient. An authentication system prevents the unauthorized injection of messages into a public channel, assuring the receiver of a message of the legitimacy of its sender.

A channel is considered public if its security is inadequate for the needs of its users. A channel such as a telephone line may therefore be considered private by some users and public by others. Any channel may be threatened with eavesdropping or injection or both, depending on its use. In telephone communication, the threat of injection is paramount, since the called party cannot determine which phone is calling. Eavesdropping, which requires the use of a wiretap, is technically more difficult and legally hazardous. In radio, by comparison, the situation is reversed. Eavesdropping is passive and involves no legal hazard, while injection exposes the illegitimate transmitter to discovery and prosecution.

Having divided our problems into those of privacy and authentication we will sometimes further subdivide authentication into message authentication, which is the problem defined above, and user authentication, in which the only task of the system is to verify that an individual is who he claims to be. For example, the identity of an individual who presents a credit card must be verified, but there is no message which he wishes to transmit. In spite of this apparent absence of a message in user authentication, the two problems are largely equivalent. In user authentication, there is an implicit message "I AM USER X," while message authentication is just verification of the identity of the party sending the message. Differences in the threat environments and other aspects of these two subproblems, however, sometimes make it convenient to distinguish between them.

Fig. 1 illustrates the flow of information in a conventional cryptographic system used for privacy of communications. There are three parties: a transmitter, a receiver, and an eavesdropper. The transmitter generates a plaintext or unenciphered message P to be communicated over an insecure channel to the legitimate receiver. In order to prevent the eavesdropper from learning P, the transmitter operates on P with an invertible transformation S_K to produce the ciphertext or cryptogram $C = S_K(P)$. The key K is transmitted only to the legitimate receiver via a secure channel, indicated by a shielded path in Fig. 1. Since the legitimate receiver knows K, he can decipher C by operating with S_K^{-1} to obtain $S_K^{-1}(C) = S_K^{-1}(S_K(P)) = P$, the original plaintext message. The secure channel cannot be used to transmit P itself for reasons of capacity or delay. For example, the secure channel might be a weekly courier and the insecure channel a telephone line.

A *cryptographic system* is a single parameter family $\{S_K\}_{K \in \{K\}}$ of invertible transformations

$$S_K : \{P\} \rightarrow \{C\} \qquad (1)$$

from a space $\{P\}$ of plaintext messages to a space $\{C\}$ of ciphertext messages. The parameter K is called the key and is selected from a finite set $\{K\}$ called the keyspace. If the message spaces $\{P\}$ and $\{C\}$ are equal, we will denote them both by $\{M\}$. When discussing individual cryptographic transformations S_K, we will sometimes omit mention of the system and merely refer to the transformation K.

The goal in designing the cryptosystem $\{S_K\}$ is to make the enciphering and deciphering operations inexpensive, but to ensure that any successful cryptanalytic operation is too complex to be economical. There are two approaches to this problem. A system which is secure due to the computational cost of cryptanalysis, but which would succumb to an attack with unlimited computation, is called *computationally secure*; while a system which can resist any cryptanalytic attack, no matter how much computation is allowed, is called *unconditionally secure*. Unconditionally secure systems are discussed in [3] and [4] and belong to that portion of information theory, called the Shannon theory, which is concerned with optimal performance obtainable with unlimited computation.

Unconditional security results from the existence of multiple meaningful solutions to a cryptogram. For example, the simple substitution cryptogram *XMD* resulting from English text can represent the plaintext messages: now, and, the, etc. A computationally secure cryptogram, in contrast, contains sufficient information to uniquely determine the plaintext and the key. Its security resides solely in the cost of computing them.

The only unconditionally secure system in common use is the *one time pad*, in which the plaintext is combined with a randomly chosen key of the same length. While such a system is provably secure, the large amount of key required makes it impractical for most applications. Except as otherwise noted, this paper deals with computationally secure systems since these are more generally applicable. When we talk about the need to develop provably secure cryptosystems we exclude those, such as the one time pad, which are unwieldly to use. Rather, we have in mind systems using only a few hundred bits of key and implementable in either a small amount of digital hardware or a few hundred lines of software.

We will call a task *computationally infeasible* if its cost as measured by either the amount of memory used or the runtime is finite but impossibly large.

Much as error correcting codes are divided into convolutional and block codes, cryptographic systems can be divided into two broad classes: *stream ciphers* and *block ciphers*. Stream ciphers process the plaintext in small chunks (bits or characters), usually producing a pseudorandom sequence of bits which is added modulo 2 to the bits of the plaintext. Block ciphers act in a purely combinatorial fashion on large blocks of text, in such a way that a small change in the input block produces a major change in the resulting output. This paper deals primarily with block ciphers, because this *error propagation* property is valuable in many authentication applications.

In an authentication system, cryptography is used to guarantee the authenticity of the message to the receiver. Not only must a meddler be prevented from injecting totally new, authentic looking messages into a channel, but he must be prevented from creating apparently authentic messages by combining, or merely repeating, old messages which he has copied in the past. A cryptographic system intended to guarantee privacy will not, in general, prevent this latter form of mischief.

To guarantee the authenticity of a message, information is added which is a function not only of the message and a secret key, but of the date and time as well; for example, by attaching the date and time to each message and encrypting the entire sequence. This assures that only someone who possesses the key can generate a message which, when decrypted, will contain the proper date and time. Care must be taken, however, to use a system in which small changes in the ciphertext result in large changes in the deciphered plaintext. This intentional error propagation ensures that if the deliberate injection of noise on the channel changes a message such as "erase file 7" into a different message such as "erase file 8," it will also corrupt the authentication information. The message will then be rejected as inauthentic.

The first step in assessing the adequacy of cryptographic systems is to classify the threats to which they are to be subjected. The following threats may occur to cryptographic systems employed for either privacy or authentication.

A *ciphertext only attack* is a cryptanalytic attack in which the cryptanalyst possesses only ciphertext.

A *known plaintext attack* is a cryptanalytic attack in which the cryptanalyst possesses a substantial quantity of corresponding plaintext and ciphertext.

A *chosen plaintext attack* is a cryptanalytic attack in which the cryptanalyst can submit an unlimited number of plaintext messages of his own choosing and examine the resulting cryptograms.

In all cases it is assumed that the opponent knows the general system $\{S_K\}$ in use since this information can be obtained by studying a cryptographic device. While many users of cryptography attempt to keep their equipment secret, many commercial applications require not only that the general system be public but that it be standard.

A ciphertext only attack occurs frequently in practice. The cryptanalyst uses only knowledge of the statistical properties of the language in use (e.g., in English, the letter e occurs 13 percent of the time) and knowledge of certain "probable" words (e.g., a letter probably begins "Dear Sir:"). It is the weakest threat to which a system can be subjected, and any system which succumbs to it is considered totally insecure.

A sy
tack fr
sages s
tion. T
tem's u
produc
encryp
tions in
of mar
plainte
frequer
conside
A ch
practic
ting a
pherin
which i
its user
plant n
For
approp
threats
workin
the ass
tems w
attack
text or
As is
system
chosen
system
subject
as auto
to buil
identif
The
tack, t
develo
foe" sy
militar
planes
challen
encryp
the rad
encrypt
a frienc
ritory.
amine t
the aut
plainte
counter
need n
Ther
cannot
which
introd
receive
ion n

combi-
ay that
change
ily with
perty is

used to
eceiver.
ting to-
nel, but
uthentic
essages
system
prevent

rmation
age and
xample,
and en-
at only
nessage
ate and
stem in
in large
al error
of noise
e 7" into
lso cor-
age will

graphic
re to be
crypto-
uthenti-

ttack in

ttack in
quantity

ttack in
number
mine the

ows the
n can be
ile many
uipment
nly that
ard.
practice.
tatistical
the letter
f certain
s "Dear
n can be
it is con

A system which is secure against a known plaintext attack frees its users from the need to keep their past messages secret, or to paraphrase them prior to declassification. This is an unreasonable burden to place on the system's users, particularly in commercial situations where product announcements or press releases may be sent in encrypted form for later public disclosure. Similar situations in diplomatic correspondence have led to the cracking of many supposedly secure systems. While a known plaintext attack is not always possible, its occurrence is frequent enough that a system which cannot resist it is not considered secure.

A chosen plaintext attack is difficult to achieve in practice, but can be approximated. For example, submitting a proposal to a competitor may result in his enciphering it for transmission to his headquarters. A cipher which is secure against a chosen plaintext attack thus frees its users from concern over whether their opponents can plant messages in their system.

For the purpose of certifying systems as secure, it is appropriate to consider the more formidable cryptanalytic threats as these not only give more realistic models of the working environment of a cryptographic system, but make the assessment of the system's strength easier. Many systems which are difficult to analyze using a ciphertext only attack can be ruled out immediately under known plaintext or chosen plaintext attacks.

As is clear from these definitions, cryptanalysis is a system identification problem. The known plaintext and chosen plaintext attacks correspond to passive and active system identification problems, respectively. Unlike many subjects in which system identification is considered, such as automatic fault diagnosis, the goal in cryptography is to build systems which are difficult, rather than easy, to identify.

The chosen plaintext attack is often called an IFF attack, terminology which descends from its origin in the development of cryptographic "identification friend or foe" systems after World War II. An IFF system enables military radars to distinguish between friendly and enemy planes automatically. The radar sends a time-varying challenge to the airplane which receives the challenge, encrypts it under the appropriate key, and sends it back to the radar. By comparing this response with a correctly encrypted version of the challenge, the radar can recognize a friendly aircraft. While the aircraft are over enemy territory, enemy cryptanalysts can send challenges and examine the encrypted responses in an attempt to determine the authentication key in use, thus mounting a chosen plaintext attack on the system. In practice, this threat is countered by restricting the form of the challenges, which need not be unpredictable, but only nonrepeating.

There are other threats to authentication systems which cannot be treated by conventional cryptography, and which require recourse to the new ideas and techniques introduced in this paper. The *threat of compromise of the receiver's authentication data* is motivated by the situation in multiuser networks where the receiver is often the system itself. The receiver's password tables and other authentication data are then more vulnerable to theft than those of the transmitter (an individual user). As shown later, some techniques for protecting against this threat also protect against the *threat of dispute*. That is, a message may be sent but later repudiated by either the transmitter or the receiver. Or, it may be alleged by either party that a message was sent when in fact none was. Unforgeable digital signatures and receipts are needed. For example, a dishonest stockbroker might try to cover up unauthorized buying and selling for personal gain by forging orders from clients, or a client might disclaim an order actually authorized by him but which he later sees will cause a loss. We will introduce concepts which allow the receiver to verify the authenticity of a message, but prevent him from generating apparently authentic messages, thereby protecting against both the threat of compromise of the receiver's authentication data and the threat of dispute.

III. Public Key Cryptography

As shown in Fig. 1, cryptography has been a derivative security measure. Once a secure channel exists along which keys can be transmitted, the security can be extended to other channels of higher bandwidth or smaller delay by encrypting the messages sent on them. The effect has been to limit the use of cryptography to communications among people who have made prior preparation for cryptographic security.

In order to develop large, secure, telecommunications systems, this must be changed. A large number of users n results in an even larger number, $(n^2 - n)/2$ potential pairs who may wish to communicate privately from all others. It is unrealistic to assume either that a pair of users with no prior acquaintance will be able to wait for a key to be sent by some secure physical means, or that keys for all $(n^2 - n)/2$ pairs can be arranged in advance. In another paper [5], the authors have considered a conservative approach requiring no new development in cryptography itself, but this involves diminished security, inconvenience, and restriction of the network to a starlike configuration with respect to initial connection protocol.

We propose that it is possible to develop systems of the type shown in Fig. 2, in which two parties communicating solely over a public channel and using only publicly known techniques can create a secure connection. We examine two approaches to this problem, called public key cryptosys-

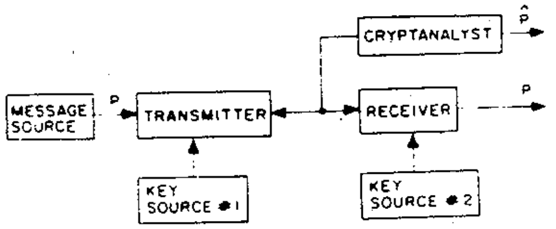


Fig. 2 Flow of information in public key system.

tems and public key distribution systems, respectively. The first are more powerful, lending themselves to the solution of the authentication problems treated in the next section, while the second are much closer to realization.

A *public key cryptosystem* is a pair of families $\{E_K\}_{K \in \{K\}}$ and $\{D_K\}_{K \in \{K\}}$ of algorithms representing invertible transformations,

$$E_K:\{M\} \to \{M\} \tag{2}$$

$$D_K:\{M\} \to \{M\} \tag{3}$$

on a finite message space $\{M\}$, such that

1) for every $K \in \{K\}$, E_K is the inverse of D_K,
2) for every $K \in \{K\}$ and $M \in \{M\}$, the algorithms E_K and D_K are easy to compute,
3) for almost every $K \in \{K\}$, each easily computed algorithm equivalent to D_K is computationally infeasible to derive from E_K,
4) for every $K \in \{K\}$, it is feasible to compute inverse pairs E_K and D_K from K.

Because of the third property, a user's enciphering key E_K can be made public without compromising the security of his secret deciphering key D_K. The cryptographic system is therefore split into two parts, a family of enciphering transformations and a family of deciphering transformations in such a way that, given a member of one family, it is infeasible to find the corresponding member of the other.

The fourth property guarantees that there is a feasible way of computing corresponding pairs of inverse transformations when no constraint is placed on what either the enciphering or deciphering transformation is to be. In practice, the cryptoequipment must contain a true random number generator (e.g., a noisy diode) for generating K, together with an algorithm for generating the $E_K - D_K$ pair from its outputs.

Given a system of this kind, the problem of key distribution is vastly simplified. Each user generates a pair of inverse transformations, E and D, at his terminal. The deciphering transformation D must be kept secret, but need never be communicated on any channel. The enciphering key E can be made public by placing it in a public directory along with the user's name and address. Anyone can then encrypt messages and send them to the user, but no one else can decipher messages intended for him. Public key cryptosystems can thus be regarded as *multiple access ciphers*.

It is crucial that the public file of enciphering keys be protected from unauthorized modification. This task is made easier by the public nature of the file. Read protection is unnecessary and, since the file is modified infrequently, elaborate write protection mechanisms can be economically employed.

A suggestive, although unfortunately useless, example of a public key cryptosystem is to encipher the plaintext, represented as a binary n-vector $\mathbf{m}$, by multiplying it by an invertible binary $n \times n$ matrix E. The cryptogram thus equals $E\mathbf{m}$. Letting $D = E^{-1}$ we have $\mathbf{m} = D\mathbf{c}$. Thus, both enciphering and deciphering require about n^2 operations. Calculation of D from E, however, involves a matrix inversion which is a harder problem. And it is at least conceptually simpler to obtain an arbitrary pair of inverse matrices than it is to invert a given matrix. Start with the identity matrix I and do elementary row and column operations to obtain an arbitrary invertible matrix E. Then starting with I do the inverses of these same elementary operations in reverse order to obtain $D = E^{-1}$. The sequence of elementary operations could be easily determined from a random bit string.

Unfortunately, matrix inversion takes only about n^3 operations. The ratio of "cryptanalytic" time (i.e., computing D from E) to enciphering or deciphering time is thus at most n, and enormous block sizes would be required to obtain ratios of 10^6 or greater. Also, it does not appear that knowledge of the elementary operations used to obtain E from I greatly reduces the time for computing D. And, since there is no round-off error in binary arithmetic, numerical stability is unimportant in the matrix inversion. In spite of its lack of practical utility, this matrix example is still useful for clarifying the relationships necessary in a public key cryptosystem.

A more practical approach to finding a pair of easily computed inverse algorithms E and D; such that D is hard to infer from E, makes use of the difficulty of analyzing programs in low level languages. Anyone who has tried to determine what operation is accomplished by someone else's machine language program knows that E itself (i.e., what E does) can be hard to infer from an algorithm for E. If the program were to be made purposefully confusing through addition of unneeded variables and statements, then determining an inverse algorithm could be made very difficult. Of course, E must be complicated enough to prevent its identification from input-output pairs.

Essentially what is required is a one-way compiler: one which takes an easily understood program written in a high level language and translates it into an incomprehensible program in some machine language. The compiler is one-way because it must be feasible to do the compilation, but infeasible to reverse the process. Since efficiency in size of program and run time are not crucial in this application, such compilers may be possible if the structure of the machine language can be optimized to assist in the confusion.

Merkle [1] has independently studied the problem of distributing keys over an insecure channel. His approach is different from that of the public key cryptosystems suggested above, and will be termed a *public key distribution system*. The goal is for two users, A and B, to securely exchange a key over an insecure channel. This key is then used by both users in a normal cryptosystem for both enciphering and deciphering. Merkle has a solution whose cryptanalytic cost grows as n^2 where n is the cost to the legitimate users. Unfortunately the cost to the legitimate users of the system is as much in transmission time as in computation, because Merkle's protocol requires potential keys to be transmitted before one key can be decided on. Merkle notes that this high transmission overhead prevents the system from being very useful in practice. If a one megabit limit is placed on the setup protocol's overhead, his technique can achieve cost ratios of approximately 10 000 to 1, which are too small for most applications. If inexpensive, high bandwidth data links become available, ratios of a million to one or greater could be achieved and the system would be of substantial practical value.

We now suggest a new public key distribution system which has several advantages. First, it requires only one "key" to be exchanged. Second, the cryptanalytic effort appears to grow exponentially in the effort of the legitimate users. And, third, its use can be tied to a public file of user information which serves to authenticate user A to user B and vice versa. By making the public file essentially a read only memory, one personal appearance allows a user to authenticate his identity many times to many users. Merkle's technique requires A and B to verify each other's identities through other means.

The new technique makes use of the apparent difficulty of computing logarithms over a finite field $GF(q)$ with a prime number q of elements. Let

$$Y = \alpha^X \bmod q, \qquad \text{for } 1 \leq X \leq q-1, \tag{4}$$

where α is a fixed primitive element of $GF(q)$, then X is referred to as the logarithm of Y to the base α, mod q:

$$X = \log_\alpha Y \bmod q, \qquad \text{for } 1 \leq Y \leq q-1. \tag{5}$$

Calculation of Y from X is easy, taking at most $2 \times \log_2 q$ multiplications [6, pp. 398–422]. For example, for $X = 18$,

$$Y = \alpha^{18} = (((\alpha^2)^2)^2)^2 \times \alpha^2. \tag{6}$$

Computing X from Y, on the other hand can be much more difficult and, for certain carefully chosen values of q, requires on the order of $q^{1/2}$ operations, using the best known algorithm [7, pp. 9, 575–576], [8].

The security of our technique depends crucially on the difficulty of computing logarithms mod q, and if an algorithm whose complexity grew as $\log_2 q$ were to be found, our system would be broken. While the simplicity of the problem statement might allow such simple algorithms, it might instead allow a proof of the problem's difficulty. For now we assume that the best known algorithm for computing logs mod q is in fact close to optimal and hence that $q^{1/2}$ is a good measure of the problem's complexity, for a properly chosen q.

Each user generates an independent random number X_i chosen uniformly from the set of integers $\{1, 2, \cdots, q-1\}$. Each keeps X_i secret, but places

$$Y_i = \alpha^{X_i} \bmod q \tag{7}$$

in a public file with his name and address. When users i and j wish to communicate privately, they use

$$K_{ij} = \alpha^{X_i X_j} \bmod q \tag{8}$$

as their key. User i obtains K_{ij} by obtaining Y_j from the public file and letting

$$K_{ij} = Y_j^{X_i} \bmod q \tag{9}$$

$$= (\alpha^{X_j})^{X_i} \bmod q \tag{10}$$

$$= \alpha^{X_j X_i} = \alpha^{X_i X_j} \bmod q. \tag{11}$$

User j obtains K_{ij} in the similar fashion

$$K_{ij} = Y_i^{X_j} \bmod q. \tag{12}$$

Another user must compute K_{ij} from Y_i and Y_j, for example, by computing

$$K_{ij} = Y_i^{(\log_\alpha Y_j)} \bmod q. \tag{13}$$

We thus see that if logs mod q are easily computed the system can be broken. While we do not currently have a proof of the converse (i.e., that the system is secure if logs mod q are difficult to compute), neither do we see any way to compute K_{ij} from Y_i and Y_j without first obtaining either X_i or X_j.

If q is a prime slightly less than 2^b, then all quantities are representable as b bit numbers. Exponentiation then takes at most $2b$ multiplications mod q, while by hypothesis taking logs requires $q^{1/2} = 2^{b/2}$ operations. The cryptanalytic effort therefore grows exponentially relative to legitimate efforts. If $b = 200$, then at most 400 multiplications are required to compute Y_i from X_i, or K_{ij} from Y_i and X_j, yet taking logs mod q requires 2^{100} or approximately 10^{30} operations.

IV. One-Way Authentication

The problem of authentication is perhaps an even more serious barrier to the universal adoption of telecommunications for business transactions than the problem of key distribution. Authentication is at the heart of any system involving contracts and billing. Without it, business cannot function. Current electronic authentication systems cannot meet the need for a purely digital, unforgeable, message dependent signature. They provide protection against third party forgeries, but do not protect against disputes between transmitter and receiver.

In order to develop a system capable of replacing the current written contract with some purely electronic form of communication, we must discover a digital phenomenon with the same properties as a written signature. It must be easy for anyone to recognize the signature as authentic, but impossible for anyone other than the legitimate signer to produce it. We will call any such technique *one-way authentication*. Since any digital signal can be copied precisely, a true digital signature must be recognizable without being known.

Consider the "login" problem in a multiuser computer system. When setting up his account, the user chooses a password which is entered into the system's password directory. Each time he logs in, the user is again asked to provide his password. By keeping this password secret from all other users, forged logins are prevented. This, however, makes it vital to preserve the security of the password directory since the information it contains would allow perfect impersonation of any user. The problem is further compounded if system operators have legitimate reasons for accessing the directory. Allowing such legitimate accesses, but preventing all others, is next to impossible.

This leads to the apparently impossible requirement for a new login procedure capable of judging the authenticity of passwords without actually knowing them. While appearing to be a logical impossibility, this proposal is easily satisfied. When the user first enters his password PW, the computer automatically and transparently computes a function $f(PW)$ and stores this, not PW, in the password directory. At each successive login, the computer calculates $f(X)$, where X is the proffered password, and compares $f(X)$ with the stored value $f(PW)$. If and only if they are equal, the user is accepted as being authentic. Since the function f must be calculated once per login, its computation time must be small. A million instructions (costing approximately $0.10 at bicentennial prices) seems to be a reasonable limit on this computation. If we could ensure, however, that calculation of f^{-1} required 10^{30} or more instructions, someone who had subverted the system to obtain the password directory could not in practice obtain PW from $f(PW)$, and could thus not perform an unauthorized login. Note that $f(PW)$ is not accepted as a password by the login program since it will automatically compute $f(f(PW))$ which will not match the entry $f(PW)$ in the password directory.

We assume that the function f is public information, so that it is not ignorance of f which makes calculation of f^{-1} difficult. Such functions are called one-way functions and were first employed for use in login procedures by R. M. Needham [9, p. 91]. They are also discussed in two recent papers [10], [11] which suggest interesting approaches to the design of one-way functions.

More precisely, a function f is a *one-way function* if, for any argument x in the domain of f, it is easy to compute the corresponding value $f(x)$, yet, for almost all y in the range of f, it is computationally infeasible to solve the equation $y = f(x)$ for any suitable argument x.

It is important to note that we are defining a function which is not invertible from a computational point of view, but whose noninvertibility is entirely different from that normally encountered in mathematics. A function f is normally called "noninvertible" when the inverse of a point y is not unique, (i.e., there exist distinct points x_1 and x_2 such that $f(x_1) = y = f(x_2)$). We emphasize that this is not the sort of inversion difficulty that is required. Rather, it must be overwhelmingly difficult, given a value y and knowledge of f, to calculate any x whatsoever with the property that $f(x) = y$. Indeed, if f is noninvertible in the usual sense, it may make the task of finding an inverse image easier. In the extreme, if $f(x) \equiv y_0$ for all x in the domain, then the range of f is $\{y_0\}$, and we can take any x as $f^{-1}(y_0)$. It is therefore necessary that f not be too degenerate. A small degree of degeneracy is tolerable and, as discussed later, is probably present in the most promising class of one-way functions.

Polynomials offer an elementary example of one-way functions. It is much harder to find a root x_0 of the polynomial equation $p(x) = y$ than it is to evaluate the polynomial $p(x)$ at $x = x_0$. Purdy [11] has suggested the use of sparse polynomials of very high degree over finite fields, which appear to have very high ratios of solution to evaluation time. The theoretical basis for one-way functions is discussed at greater length in Section VI. And, as shown in Section V, one-way functions are easy to devise in practice.

The one-way function login protocol solves only some of the problems arising in a multiuser system. It protects against compromise of the system's authentication data when it is not in use, but still requires the user to send the true password to the system. Protection against eavesdropping must be provided by additional encryption, and protection against the threat of dispute is absent altogether.

A public key cryptosystem can be used to produce a true one-way authentication system as follows. If user A wishes to send a message M to user B, he "deciphers" it in his secret deciphering key and sends $D_A(M)$. When user B receives it, he can read it, and be assured of its authenticity by "enciphering" it with user A's public enciphering key E_A. B also saves $D_A(M)$ as proof that the message came from A. Anyone can check this claim by operating on $D_A(M)$ with the publicly known operation E_A to recover M. Since only A could have generated a message with this property, the solution to the one-way authentication problem would follow immediately from the development of public key cryptosystems.

One-way message authentication has a partial solution suggested to the authors by Leslie Lamport of Massachusetts Computer Associates. This technique employs a one-way function f mapping k-dimensional binary space into itself for k on the order of 100. If the transmitter wishes to send an N bit message he generates $2N$, randomly chosen, k-dimensional binary vectors $x_1,X_1,x_2,X_2,\cdots,x_N,X_N$ which he keeps secret. The receiver is given the corresponding images under f, namely $y_1,Y_1,y_2,Y_2,\cdots,y_N,Y_N$. Later, when the message $\boldsymbol{m} = (m_1,m_2,\cdots,m_N)$ is to be sent, the transmitter sends x_1 or X_1 depending on whether $m_1 = 0$ or 1. He sends x_2 or X_2 depending on whether $m_2 = 0$ or 1, etc. The receiver operates with f on the first received block and sees whether it yields y_1 or Y_1 as its image and thus learns whether it was x_1 or X_1, and whether $m_1 = 0$ or 1. In a similar manner the receiver is able to determine $m_2,m_3,\cdots,m_N$. But the receiver is incapable of forging a change in even one bit of $\boldsymbol{m}$.

This is only a partial solution because of the approximately 100-fold data expansion required. There is, however, a modification which eliminates the expansion problem when N is roughly a megabit or more. Let g be a one-way mapping from binary N-space to binary n-space where n is approximately 50. Take the N bit message $\boldsymbol{m}$

mising
ie-way
: poly-
: poly-
use of
fields,
› eval-
ctions
shown
/ise in
' some
'otects
n data
nd the
eaves-
›n, and
t alto-
: a true
wishes
: in his
user B
enticity
ng key
e came
ing on
'ecover
ith this
ication
›pment
olution
Massa-
ploys a
y space
smitter
N, ran-
vectors
The re-
namely
;e m =
ds x_1 or
$_2$ or X_2
iver op-
whether
er it was
ıner the
: the re-
ıe bit of
approxi-
is, how-
:pansion
et g be a
n-space
›ssage m

and operate on it with g to obtain the n bit vector m'. Then use the previous scheme to send m'. If $N = 10^6$, $n = 50$, and $k = 100$, this adds $kn = 5000$ authentication bits to the message. It thus entails only a 5 percent data expansion during transmission (or 15 percent if the initial exchange of $y_1, Y_1, \cdots, y_N, Y_N$ is included). Even though there are a large number of other messages (2^{N-n} on the average) with the same authentication sequence, the one-wayness of g makes them computationally infeasible to find and thus to forge. Actually g must be somewhat stronger than a normal one-way function, since an opponent has not only m' but also one of its inverse images m. It must be hard even given m to find a different inverse image of m'. Finding such functions appears to offer little trouble (see Section V).

There is another partial solution to the one-way user authentication problem. The user generates a password X which he keeps secret. He gives the system $f^T(X)$, where f is a one-way function. At time t the appropriate authenticator is $f^{T-t}(X)$, which can be checked by the system by applying $f^t(X)$. Because of the one-wayness of f, past responses are of no value in forging a new response. The problem with this solution is that it can require a fair amount of computation for legitimate login (although many orders of magnitude less than for forgery). If for example t is incremented every second and the system must work for one month on each password then $T = 2.6$ million. Both the user and the system must then iterate f an average of 1.3 million times per login. While not insurmountable, this problem obviously limits use of the technique. The problem could be overcome if a simple method for calculating $f^{(2\uparrow n)}$, for $n = 1,2, \cdots$ could be found, much as $X^8 = ((X^2)^2)^2$. For then binary decompositions of $T - t$ and t would allow rapid computation of f^{T-t} and f^t. It may be, however, that rapid computation of f^n precludes f from being one-way.

V. Problem Interrelations and Trap Doors

In this section, we will show that some of the cryptographic problems presented thus far can be reduced to others, thereby defining a loose ordering according to difficulty. We also introduce the more difficult problem of trap doors.

In Section II we showed that a cryptographic system intended for privacy can also be used to provide authentication against third party forgeries. Such a system can be used to create other cryptographic objects, as well.

A cryptosystem which is secure against a known plaintext attack can be used to produce a one-way function.

As indicated in Fig. 3, take the cryptosystem $\{S_K:\{P\} \to \{C\}\}_{K \in \{K\}}$ which is secure against a known plaintext attack, fix $P = P_0$ and consider the map

$$f:\{K\} \to \{C\} \tag{14}$$

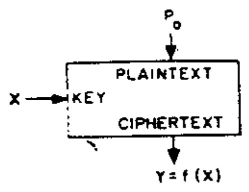


Fig. 3. Secure cryptosystem used as one-way function.

defined by

$$f(X) = S_X(P_0). \tag{15}$$

This function is one-way because solving for X given $f(X)$ is equivalent to the cryptanalytic problem of finding the key from a single known plaintext–cryptogram pair. Public knowledge of f is now equivalent to public knowledge of $\{S_K\}$ and P_0.

While the converse of this result is not necessarily true, it is possible for a function originally found in the search for one-way functions to yield a good cryptosystem. This actually happened with the discrete exponential function discussed in Section III [8].

One-way functions are basic to both block ciphers and key generators. A key generator is a pseudorandom bit generator whose output, the keystream, is added modulo 2 to a message represented in binary form, in imitation of a one-time pad. The key is used as a "seed" which determines the pseudorandom keystream sequence. A known plaintext attack thus reduces to the problem of determining the key from the keystream. For the system to be secure, computation of the key from the keystream must be computationally infeasible. While, for the system to be usable, calculation of the keystream from the key must be computationally simple. Thus a good key generator is, almost by definition, a one-way function.

Use of either type of cryptosystem as a one way function suffers from a minor problem. As noted earlier, if the function f is not uniquely invertible, it is not necessary (or possible) to find the actual value of X used. Rather any X with the same image will suffice. And, while each mapping S_K in a cryptosystem must be bijective, there is no such restriction on the function f from key to cryptogram defined above. Indeed, guaranteeing that a cryptosystem has this property appears quite difficult. In a good cryptosystem the mapping f can be expected to have the characteristics of a randomly chosen mapping (i.e., $f(X_i)$ is chosen uniformly from all possible Y, and successive choices are independent). In this case, if X is chosen uniformly and there are an equal number of keys and messages (X and Y), then the probability that the resultant Y has $k + 1$ inverses is approximately $e^{-1}/k!$ for $k = 0,1,2,3, \cdots$. This is a Poisson distribution with mean $\lambda = 1$, shifted by 1 unit. The expected number of inverses is thus only 2. While it is possible for f to be more degenerate, a good cryptosystem will not be too degenerate since then the key is not being well used. In the worst case, if $f(X) = Y_0$ for some Y_0, we have $S_K(P_0) \equiv C_0$, and encipherment of P_0 would not depend on the key at all!

While we are usually interested in functions whose domain and range are of comparable size, there are exceptions. In the previous section we required a one-way function mapping long strings onto much shorter ones. By using a block cipher whose key length is larger than the blocksize, such functions can be obtained using the above technique.

Evans *et al.* [10] have a different approach to the problem of constructing a one-way function from a block cipher. Rather than selecting a fixed P_0 as the input, they use the function

$$f(X) = S_X(X). \tag{16}$$

This is an attractive approach because equations of this form are generally difficult to solve, even when the family S is comparatively simple. This added complexity, however, destroys the equivalence between the security of the system S under a known plaintext attack and the one-wayness of f.

Another relationship has already been shown in Section IV.

A public key cryptosystem can be used to generate a one-way authentication system.

The converse does not appear to hold, making the construction of a public key cryptosystem a strictly more difficult problem than one-way authentication. Similarly, a public key cryptosystem can be used as a public key distribution system, but not conversely.

Since in a public key cryptosystem the general system in which E and D are used must be public, specifying E specifies a complete algorithm for transforming input messages into output cryptograms. As such a public key system is really a set of *trap-door one-way functions.* These are functions which are not really one-way in that simply computed inverses exist. But given an algorithm for the forward function it is computationally infeasible to find a simply computed inverse. Only through knowledge of certain *trap-door information* (e.g., the random bit string which produced the E-D pair) can one easily find the easily computed inverse.

Trap doors have already been seen in the previous paragraph in the form of *trap-door one-way functions,* but other variations exist. A *trap-door cipher* is one which strongly resists cryptanalysis by anyone not in possession of *trap-door information* used in the design of the cipher. This allows the designer to break the system after he has sold it to a client and yet falsely to maintain his reputation as a builder of secure systems. It is important to note that it is not greater cleverness or knowledge of cryptography which allows the designer to do what others cannot. If he were to lose the trap-door information he would be no better off than anyone else. The situation is precisely analogous to a combination lock. Anyone who knows the combination can do in seconds what even a skilled locksmith would require hours to accomplish. And yet, if he forgets the combination, he has no advantage.

A trap-door cryptosystem can be used to produce a public key distribution system.

For A and B to establish a common private key, A chooses a key at random and sends an arbitrary plaintext–cryptogram pair to B. B, who made the trap-door cipher public, but kept the trap-door information secret, uses the plaintext-cryptogram pair to solve for the key. A and B now have a key in common.

There is currently little evidence for the existence of trap-door ciphers. However they are a distinct possibility and should be remembered when accepting a cryptosystem from a possible opponent [12].

By definition, we will require that a trap-door problem be one in which it is computationally feasible to devise the trap door. This leaves room for yet a third type of entity for which we shall use the prefix "quasi." For example a *quasi one-way function* is not one-way in that an easily computed inverse exists. However, it is computationally infeasible even for the designer, to find the easily computed inverse. Therefore a quasi one-way function can be used in place of a one-way function with essentially no loss in security.

Losing the trap-door information to a trap-door one-way function makes it into a quasi one-way function, but there may also be one-way functions not obtainable in this manner.

It is entirely a matter of definition that quasi one-way functions are excluded from the class of one-way functions. One could instead talk of one-way functions in the wide sense or in the strict sense.

Similarly, a quasi secure cipher is a cipher which will successfully resist cryptanalysis, even by its designer, and yet for which there exists a computationally efficient cryptanalytic algorithm (which is of course computationally infeasible to find). Again, from a practical point of view, there is essentially no difference between a secure cipher and a quasi secure one.

We have already seen that public key cryptosystems imply the existence of trap-door one-way functions. However the converse is not true. For a trap-door one-way function to be usable as a public key cryptosystem, it must be invertible (i.e., have a unique inverse.)

VI. Computational Complexity

Cryptography differs from all other fields of endeavor in the ease with which its requirements may appear to be satisfied. Simple transformations will convert a legible text into an apparently meaningless jumble. The critic, who wishes to claim that meaning might yet be recovered by cryptanalysis, is then faced with an arduous demonstration if he is to prove his point of view correct. Experience has shown, however, that few systems can resist the concerted attack of skillful cryptanalysts, and many supposedly secure systems have subsequently been broken.

In consequence of this, judging the worth of new systems has always been a central concern of cryptographers. During the sixteenth and seventeenth centuries, mathematical arguments were often invoked to argue the strength of cryptographic methods, usually relying on counting methods which showed the astronomical number

DIFFIE
of poss-
be laid
gebrai
whose
ken, th
curity
certific
Duri
to swin
connec
[3] sho
in use
form
system
a key w
messag
unwiel
key cry
be unc
always
the me
the pro
search.
The
discipli
tion: co
algorit
compu
has co
studyi
gressio
plicati
one-wa
A fur
polyno
Turing
polyno
think o
it is m
must b
are pro
pp. 405
Ther
which c
techniq
unlimit
may no
(for nor
in poly
one wit
class N
questio
strictly
Amo
but not
trevelin
proposi
[illegible]
proble

key, A
ɣ plain-
door ci-
secret,
ɜ key. A
ence of
ɪsibility
ɤsystem
roblem
vise the
f entity
.mple a
ı easily
.ionally
nputed
ɔe used
loss in
ne-way
ɪt there
in this
ne-way
ɪctions.
ɪe wide
ch will
er, and
ficient
tation-
oint of
secure
vstems
ctions.
ɔe-way
ɪt must
deavor
r to be
ɪle text
c, who
red by
tration
ɪce has
.certed
dly se-
ystems
ɪphers.
nathe-
ɪe the
ing on
ımber

of possible keys. Though the problem is far too difficult to be laid to rest by such simple methods, even the noted algebraist Cardano fell into this trap [2, p. 145]. As systems whose strength had been so argued were repeatedly broken, the notion of giving mathematical proofs for the security of systems fell into disrepute and was replaced by certification via crypanalytic assault.

During this century, however, the pendulum has begun to swing back in the other direction. In a paper intimately connected with the birth of information theory, Shannon [3] showed that the one time pad system, which had been in use since the late twenties offered "perfect secrecy" (a form of unconditional security). The provably secure systems investigated by Shannon rely on the use of either a key whose length grows linearly with the length of the message or on perfect source coding and are therefore too unwieldy for most purposes. We note that neither public key cryptosystems nor one-way authentication systems can be unconditionally secure because the public information always determines the secret information uniquely among the members of a finite set. With unlimited computation, the problem could therefore be solved by a straightforward search.

The past decade has seen the rise of two closely related disciplines devoted to the study of the costs of computation: computational complexity theory and the analysis of algorithms. The former has classified known problems in computing into broad classes by difficulty, while the latter has concentrated on finding better algorithms and studying the resources they consume. After a brief digression into complexity theory, we will examine its application to cryptography, particularly the analysis of one-way functions.

A function is said to belong to the complexity class ***P*** (for polynomial) if it can be computed by a deterministic Turing Machine in a time which is bounded above by some polynomial function of the length of its input. One might think of this as the class of easily computed functions, but it is more accurate to say that a function not in this class must be hard to compute for at least some inputs. There are problems which are known not to be in the class ***P*** [13, pp. 405–425].

There are many problems which arise in engineering which cannot be solved in polynomial time by any known techniques, unless they are run on a computer with an unlimited degree of parallelism. These problems may or may not belong to the class ***P***, but belong to the class ***NP*** (for nondeterministic, polynomial) of problems solvable in polynomial time on a "nondeterministic" computer (i.e., one with an unlimited degree of parallelism). Clearly the class ***NP*** includes the class ***P***, and one of the great open questions in complexity theory is whether the class ***NP*** is strictly larger.

Among the problems known to be solvable in ***NP*** time, but not known to be solvable in ***P*** time, are versions of the traveling salesman problem, the satisfiability problem for propositional calculus, the knapsack problem, the graph coloring problem, and many scheduling and minimization problems [13, pp. 363–404], [14]. We see that it is not lack of interest or effort which has prevented people from finding solutions in ***P*** time for these problems. It is thus strongly believed that at least one of these problems must not be in the class ***P***, and that therefore the class ***NP*** is strictly larger.

Karp has identified a subclass of the ***NP*** problems, called ***NP*** complete, with the property that if any one of them is in ***P***, then all ***NP*** problems are in ***P***. Karp lists 21 problems which are ***NP*** complete, including all of the problems mentioned above [14].

While the ***NP*** complete problems show promise for cryptographic use, current understanding of their difficulty includes only worst case analysis. For cryptographic purposes, typical computational costs must be considered. If, however, we replace worst case computation time with average or typical computation time as our complexity measure, the current proofs of the equivalences among the ***NP*** complete problems are no longer valid. This suggests several interesting topics for research. The ensemble and typicality concepts familiar to information theorists have an obvious role to play.

We can now identify the position of the general cryptanalytic problem among all computational problems.

*The cryptanalytic difficulty of a system whose encryption and decryption operations can be done in **P** time cannot be greater than **NP**.*

To see this, observe that any cryptanalytic problem can be solved by finding a key, inverse image, etc., chosen from a finite set. Choose the key nondeterministically and verify in ***P*** time that it is the correct one. If there are M possible keys to choose from, an M-fold parallelism must be employed. For example in a known plaintext attack, the plaintext is encrypted simultaneously under each of the keys and compared with the cryptogram. Since, by assumption, encryption takes only ***P*** time, the cryptanalysis takes only ***NP*** time.

We also observe that the general cryptanalytic problem is ***NP*** complete. This follows from the breadth of our definition of cryptographic problems. A one-way function with an ***NP*** complete inverse will be discussed next.

Cryptography can draw directly from the theory of ***NP*** complexity by examining the way in which ***NP*** complete problems can be adapted to cryptographic use. In particular, there is an ***NP*** complete problem known as the knapsack problem which lends itself readily to the construction of a one-way function.

Let $y = f(x) = \mathbf{a} \cdot \mathbf{x}$ where $\mathbf{a}$ is a known vector of n intergers $(a_1, a_2, \cdots, a_n)$ and $\mathbf{x}$ is a binary n-vector. Calculation of y is simple, involving a sum of at most n integers. The problem of inverting f is known as the knapsack problem and requires finding a subset of the $\{a_i\}$ which sum to y.

Exhaustive search of all 2^n subsets grows exponentially and is computationally infeasible for n greater than 100 or so. Care must be exercised, however, in selecting the parameters of the problem to ensure that shortcuts are not possible. For example if $n = 100$ and each a_i is 32 bits long, y is at most 39 bits long, and f is highly degenerate; requiring on the average only 2^{38} tries to find a solution. Somewhat more trivially, if $a_i = 2^{i-1}$ then inverting f is equivalent to finding the binary decomposition of y.

This example demonstrates both the great promise and the considerable shortcomings of contemporary complexity theory. The theory only tells us that the knapsack problem is probably difficult in the worst case. There is no indication of its difficulty for any particular array. It appears, however, that choosing the $\{a_i\}$ uniformly from $\{0,1,2,\cdots,2^{n-1}\}$ results in a hard problem with probability one as $n \rightarrow \infty$.

Another potential one-way function, of interest in the analysis of algorithms, is exponentiation mod q, which was suggested to the authors by Prof. John Gill of Stanford University. The one-wayness of this functions has already been discussed in Section III.

VII. Historical Perspective

While at first the public key systems and one-way authentication systems suggested in this paper appear to be unportended by past cryptographic developments, it is possible to view them as the natural outgrowth of trends in cryptography stretching back hundreds of years.

Secrecy is at the heart of cryptography. In early cryptography, however, there was a confusion about what was to be kept secret. Cryptosystems such as the Caesar cipher (in which each letter is replaced by the one three places further on, so A is carried to D, B to E, etc.) depended for their security on keeping the entire encryption process secret. After the invention of the telegraph [2, p. 191], the distinction between a general system and a specific key allowed the general system to be compromised, for example by theft of a cryptographic device, without compromising future messages enciphered in new keys. This principle was codified by Kerchoffs [2, p. 235] who wrote in 1881 that the compromise of a cryptographic system should cause no inconvenience to the correspondents. About 1960, cryptosystems were put into service which were deemed strong enough to resist a known plaintext cryptanalytic attack, thereby eliminating the burden of keeping old messages secret. Each of these developments decreased the portion of the system which had to be protected from public knowledge, eliminating such tedious expedients as paraphrasing diplomatic dispatches before they were presented. Public key systems are a natural continuation of this trend toward decreasing secrecy.

Prior to this century, cryptographic systems were limited to calculations which could be carried out by hand or with simple slide-rule-like devices. The period immediately after World War I saw the beginning of a revolutionary trend which is now coming to fruition. Special purpose machines were developed for enciphering. Until the development of general purpose digital hardware, however, cryptography was limited to operations which could be performed with simple electromechanical systems. The development of digital computers has freed it from the limitations of computing with gears and has allowed the search for better encryption methods according to purely

The failure of numerous attempts to demonstrate the soundness of cryptographic systems by mathematical proof led to the paradigm of certification by cryptanalytic attack set down by Kerchoffs [2, p. 234] in the last century. Although some general rules have been developed, which aid the designer in avoiding obvious weaknesses, the ultimate test is an assault on the system by skilled cryptanalysts under the most favorable conditions (e.g., a chosen plaintext attack). The development of computers has led for the first time to a mathematical theory of algorithms which can begin to approach the difficult problem of estimating the computational difficulty of breaking a cryptographic system. The position of mathematical proof may thus come full circle and be reestablished as the best method of certification.

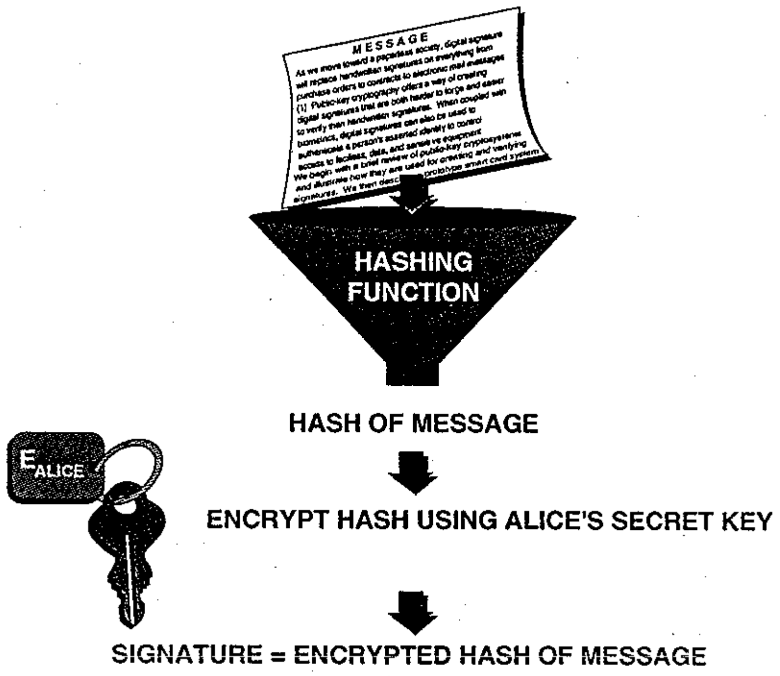
The last characteristic which we note in the history of cryptography is the division between amateur and professional cryptographers. Skill in production cryptanalysis has always been heavily on the side of the professionals, but innovation, particularly in the design of new types of cryptographic systems, has come primarily from the amateurs. Thomas Jefferson, a cryptographic amateur, invented a system which was still in use in World War II [2, pp. 192–195], while the most noted cryptographic system of the twentieth century, the rotor machine, was invented simultaneously by four separate people, all amateurs [2, pp. 415, 420, 422–424]. We hope this will inspire others to work in this fascinating area in which participation has been discouraged in the recent past by a nearly total government monopoly.

References

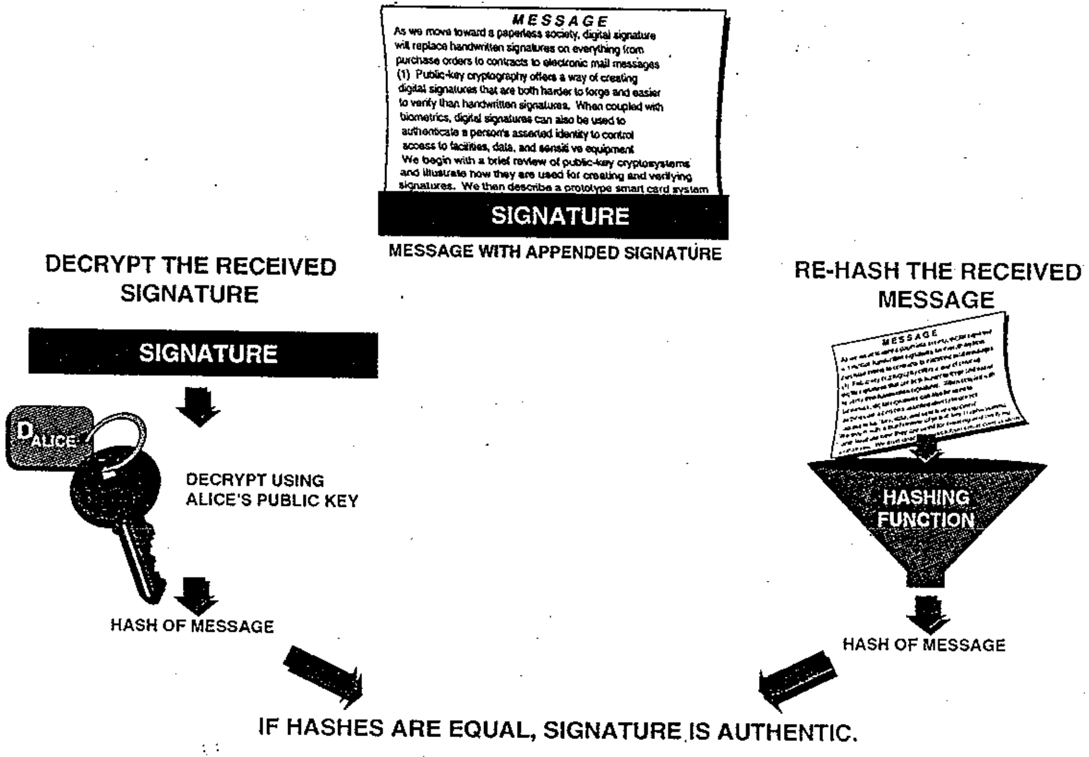
[1] R. Merkle, "Secure communication over an insecure channel," submitted to *Communications of the ACM*.
[2] D. Kahn, *The Codebreakers, The Story of Secret Writing*. New York: Macmillan, 1967.
[3] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, pp. 656–715, Oct. 1949.
[4] M. E. Hellman, "An extension of the Shannon theory approach to cryptography," submitted to *IEEE Trans. Inform. Theory*, Sept. 1975.
[5] W. Diffie and M. E. Hellman, "Multiuser cryptographic techniques," presented at National Computer Conference, New York, June 7–10, 1976.
[6] D. Knuth, *The Art of Computer Programming, Vol. 2, Semi-Numerical Algorithms*. Reading, MA.: Addison–Wesley, 1969.
[7] ——, *The Art of Computer Programming, Vol. 3, Sorting and Searching*. Reading, MA.: Addison–Wesley, 1973.
[8] S. Pohlig and M. E. Hellman, "An improved algorithm for computing algorithms in $GF(p)$ and its cryptographic significance," submitted to *IEEE Trans. Inform. Theory*.
[9] M. V. Wilkes, *Time-Sharing Computer Systems*. New York: Elsevier, 1972.
[10] A. Evans, Jr., W. Kantrowitz, and E. Weiss, "A user authentication system not requiring secrecy in the computer," *Communications of the ACM*, vol. 17, pp. 437–442, Aug. 1974.
[11] G. B. Purdy, "A high security log-in procedure," *Communications of the ACM*, vol. 17, pp. 442–445, Aug. 1974.
[12] W. Diffie and M. E. Hellman, "Cryptanalysis of the NBS data encryption standard" submitted to *Computer*, May 1976.
[13] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, MA.: Addison-Wesley, 1974.
[14] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. R. E. Miller and J. W. [illegible] New York: Plenum, 1972, pp. 85–104.

E

HOW ALICE CREATES A DIGITAL SIGNATURE


MESSAGE
HASHING FUNCTION
HASH OF MESSAGE
E_{ALICE}
ENCRYPT HASH USING ALICE'S SECRET KEY
SIGNATURE = ENCRYPTED HASH OF MESSAGE

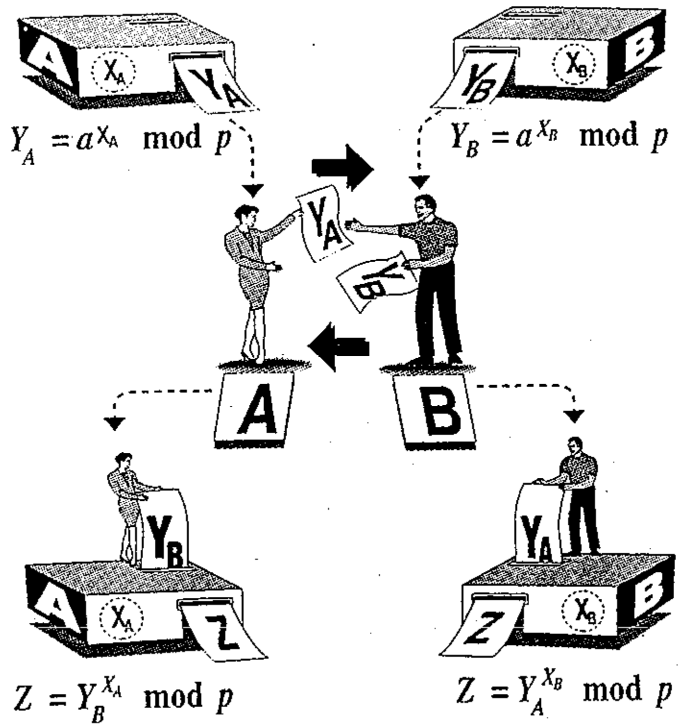
HOW BOB VERIFIES ALICE'S SIGNATURE


MESSAGE
SIGNATURE
MESSAGE WITH APPENDED SIGNATURE
DECRYPT THE RECEIVED SIGNATURE
SIGNATURE
D_{ALICE}
DECRYPT USING ALICE'S PUBLIC KEY
HASH OF MESSAGE
RE-HASH THE RECEIVED MESSAGE
MESSAGE
HASHING FUNCTION
HASH OF MESSAGE
IF HASHES ARE EQUAL, SIGNATURE IS AUTHENTIC.

As we move toward a paperless society, digital signature will replace handwritten signatures on everything from purchase orders to contracts to electronic mail messages (1) Public-key cryptography offers a way of creating digital signatures that are both harder to forge and easier to verify than handwritten signatures. When coupled with biometrics, digital signatures can also be used to authenticate a person's asserted identity to control access to facilities, data, and sensiti ve equipment We begin with a brief review of public-key cryptosystems and illustrate how they are used for creating and verifying signatures. We then describe a prototype smart card system

F

DIFFIE-HELLMAN
SECURE ELECTRONIC EXCHANGE OF KEYS



DIFFIE-HELLMAN
SECURE ELECTRONIC EXCHANGE KEYS

$$Y_B^{X_A} = a^{X_B X_A} \; mod\, p = Z$$

$$Y_A^{X_B} = a^{X_A X_B} \; mod\, p = Z$$